Karppa, Matti; Kaski, Petteri

Probabilistic tensors and opportunistic boolean matrix multiplication

Published in:
Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms

Published: 01/01/2019

Document Version
Publisher's PDF, also known as Version of record

Please cite the original version:
Karppa, M., & Kaski, P. (2019). Probabilistic tensors and opportunistic boolean matrix multiplication. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms* (pp. 496-515)
https://epubs.siam.org/doi/abs/10.1137/1.9781611975482.31?mobileUi=0&

Probabilistic Tensors and Opportunistic Boolean Matrix Multiplication*

Matti Karppa† Petteri Kaski†

Abstract

We introduce probabilistic extensions of classical deterministic measures of algebraic complexity of a tensor, such as the rank and the border rank. We show that these probabilistic extensions satisfy various natural and algorithmically serendipitous properties, such as submultiplicativity under taking of Kronecker products. Furthermore, the probabilistic extensions enable improvements over their deterministic counterparts for specific tensors of interest, starting from the tensor $\langle 2,2,2\rangle$ that represents 2×2 matrix multiplication. While it is well known that the (deterministic) tensor rank and border rank satisfy

$$\operatorname{rk}\langle 2,2,2\rangle = 7 \qquad \text{and} \qquad \underline{\operatorname{rk}}\langle 2,2,2\rangle = 7$$

[V. Strassen, *Numer. Math.* 13 (1969); J. E. Hopcroft and L. R. Kerr, *SIAM J. Appl. Math.* 20 (1971); S. Winograd, *Linear Algebra Appl.* 4 (1971); J. M. Landsberg, *J. AMS* 19 (2006)], we show that the *probabilistic* tensor rank and border rank satisfy

$$\widetilde{\operatorname{rk}}\langle 2,2,2\rangle \leq 6 + \frac{6}{7} \qquad \text{and} \qquad \underline{\widetilde{\operatorname{rk}}}\langle 2,2,2\rangle \leq 6 + \frac{2}{3}\,.$$

By submultiplicativity, this leads immediately to novel randomized algorithm designs, such as algorithms for Boolean matrix multiplication as well as detecting and estimating the number of triangles in graphs.

Our algorithms are *opportunistic* in the sense that their worst-case scaling is essentially governed by the probabilistic rank, yet their result is accumulated through independent repetitions, where the partial result can be inspected at each repeat for possible early termination, and each repeat scales according to the rank of the outcome-tensors. For example, representing $\langle 2,2,2\rangle$ probabilistically using an ensemble of tensors of rank 6, we obtain an algorithm that, with high probability, multiplies two $2^d \times 2^d$ Boolean matrices in $\tilde{O}((6+\frac{6}{7})^d)$ operations. This algorithm consists of independent repeats that each run in $O(6^d)$ operations and enable inspection of the partial result at each repeat. Analogously, a probabilistic representation of $\langle 2,2,2\rangle$ using tensors of border rank 5 gives an algorithm that runs in $\tilde{O}((6+\frac{2}{3})^d)$ operations, consisting of repeats that run in $\tilde{O}(5^d)$ operations each.

Asymptotically, we use Adleman's argument to show that, over the complex field, the support rank exponent ω_{s} of matrix multiplication [H. Cohn and C. Umans, *SODA*'12] gives the lower bound $\omega_{\mathrm{s}} \leq \inf\{\tau : \widetilde{\operatorname{rk}}\langle t,t,t\rangle = O(t^\tau)\}$ for probabilistic tensor rank. While this enables an approach to obtaining asymptotically faster algorithm designs for matrix multiplication via the Cohn–Umans inequality $\omega \leq \frac{3}{2}\omega_{\mathrm{s}} - 1$, the main motivation for the present paper is to enable an approach towards fast practical algorithms using small probabilistic tensors.

1 Introduction.

The problem of multiplying two given $n \times n$ matrices with entries over the Boolean algebra $\mathbb{B} = (0, 1, \vee, \wedge)$ is a fundamental combinatorial problem which enables fastest known algorithms for a large number of combinatorial problems, such as transitive closure, context-free parsing, and triangle detection [35, 37, 45, 71, 73]. Yet progress in fast algorithms for Boolean matrix multiplication has been confined to essentially two lines of study. The first line of study embeds the Boolean entries to a ring, and employs techniques for fast matrix multiplication over rings to recover the result. Currently the fastest such algorithms run in time $O(n^{\omega+o(1)})$, where $2 \leq \omega < 2.3728639$ is the exponent of matrix multiplication [29, 33, 72, 56]. While such algorithms are the fastest known in terms of asymptotic efficiency, practically efficient algorithms for fast matrix multiplication over rings remain limited to recursive approaches using small base tensors (e.g. [7, 12, 22, 32, 38, 42, 43, 49, 51, 57, 58]) or trilinear aggregation–cancellation techniques [60, 61] for a small number of independent matrix multiplications (e.g. [47, 48]). This situation withstanding, a second line of study seeks advanced combinatorial techniques without the use of tensors [6, 11, 23, 73, 76]. Currently, the fastest such algorithm runs in time $\hat{O}(n^3/\log^4 n)$ [76].

This paper seeks further progress in the study of Boolean matrix multiplication. In essence, we develop probabilistic extensions of existing tensor techniques to enable us to characterize which tensor-based fast matrix multiplication algorithms can be *broken* to yield, through randomization, a *faster* algorithm for Boolean matrix multiplication.

We start with an example that illustrates the key algorithmic ideas by breaking and randomizing the Strassen–Winograd algorithm.

*The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement 338077 "Theory and Practice of Advanced Search and Enumeration". We gratefully acknowledge the use of computational resources provided by the Aalto Science-IT project at Aalto University.

†Department of Computer Science, Aalto University, Helsinki, Finland. E-mail: firstname.lastname@aalto.fi

1.1 Breaking and Randomizing. The Strassen–Winograd algorithm [75] is an addition-optimal[1] [19, 64] variant of Strassen's seminal algorithm [69] for multiplying two $n \times n$ matrices in $O(n^{\log_2 7})$ operations over a ring. More precisely, the Strassen–Winograd algorithm performs a 2×2 matrix multiplication

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

using 15 additions (including subtractions) and 7 multiplications:[2]

$$\begin{array}{ll} T_1 = A_{21} + A_{22} & S_1 = B_{21} + B_{22} \\ T_2 = A_{12} & S_2 = B_{21} \\ T_3 = A_{12} + A_{22} & S_3 = B_{12} - B_{22} \\ T_4 = A_{21} + T_3 & S_4 = B_{21} - S_3 \\ T_5 = A_{11} + T_4 & S_5 = B_{12} \\ T_6 = A_{21} & S_6 = B_{11} - S_4 \\ T_* = A_{11} & S_* = B_{11} \end{array}$$

$$\begin{array}{ll} Q_1 = T_1 \cdot S_1 & U_1 = Q_2 - Q_4 \\ Q_2 = T_2 \cdot S_2 & U_2 = U_1 - Q_3 \\ Q_3 = T_3 \cdot S_3 & U_3 = Q_5 - U_1 \\ Q_4 = T_4 \cdot S_4 & C_{11} = Q_2 + Q_* \\ Q_5 = T_5 \cdot S_5 & C_{12} = U_3 - Q_1 \\ Q_6 = T_6 \cdot S_6 & C_{21} = Q_6 - U_2 \\ Q_* = T_* \cdot S_* & C_{22} = Q_1 + U_2 \end{array} \quad . \tag{1}$$

Simplifying (1), we readily verify that

$$\begin{array}{ll} C_{11} = A_{11}B_{11} + A_{12}B_{21} & C_{12} = A_{11}B_{12} + A_{12}B_{22} \\ C_{21} = A_{21}B_{11} + A_{22}B_{21} & C_{22} = A_{21}B_{12} + A_{22}B_{22} \end{array} \quad . \tag{2}$$

Used recursively, the Strassen–Winograd algorithm makes exactly $5(7^d - 4^d)$ additions and 7^d multiplications of scalars to multiply two $2^d \times 2^d$ matrices. The Strassen–Winograd algorithm and its variants are also among the most efficient implementations for multiplying large matrices in practice, both for shared-memory and distributed-memory architectures [7, 12, 22, 32, 38, 42, 43, 49, 51, 57, 58].

Breaking the algorithm. Let us now break the Strassen–Winograd algorithm. Observe the terms highlighted with an asterisk-subscript "$*$" in (1). Let us remove these terms. This leaves us with only 14 additions and 6 multiplications, and a clearly broken matrix multiplication algorithm:

$$\begin{array}{ll} T_1 = A_{21} + A_{22} & S_1 = B_{21} + B_{22} \\ T_2 = A_{12} & S_2 = B_{21} \\ T_3 = A_{12} + A_{22} & S_3 = B_{12} - B_{22} \\ T_4 = A_{21} + T_3 & S_4 = B_{21} - S_3 \\ T_5 = A_{11} + T_4 & S_5 = B_{12} \\ T_6 = A_{21} & S_6 = B_{11} - S_4 \end{array}$$

$$\begin{array}{ll} Q_1 = T_1 \cdot S_1 & U_1 = Q_2 - Q_4 \\ Q_2 = T_2 \cdot S_2 & U_2 = U_1 - Q_3 \\ Q_3 = T_3 \cdot S_3 & U_3 = Q_5 - U_1 \\ Q_4 = T_4 \cdot S_4 & C_{11} = Q_2 \\ Q_5 = T_5 \cdot S_5 & C_{12} = U_3 - Q_1 \\ Q_6 = T_6 \cdot S_6 & C_{21} = Q_6 - U_2 \\ & C_{22} = Q_1 + U_2 \end{array} \quad . \tag{3}$$

Indeed, (3) simplifies to

$$\begin{array}{ll} C_{11} = A_{12}B_{21} & C_{12} = A_{11}B_{12} + A_{12}B_{22} \\ C_{21} = A_{21}B_{11} + A_{22}B_{21} & C_{22} = A_{21}B_{12} + A_{22}B_{22} \end{array} \quad . \tag{4}$$

But the only defect in (4) is the absence of the product $A_{11}B_{11}$. That is, 7 out of the 8 products are present, *at the cost of only* 6 *multiplications*.

Used recursively, the algorithm (3) yields a recursively broken result, with exactly 7^d out of the required 8^d products present in the broken result, using exactly $7(6^d - 4^d)$ additions and 6^d multiplications of scalars in the process. That is, the fraction of present products is $(\frac{7}{8})^d$, and these products can be obtained in only $8 \cdot 6^d - 7 \cdot 4^d = O(6^d)$ operations.

Randomization. Let us now randomize the broken algorithm, with the objective of randomizing which 7^d products among the 8^d products are present in the broken result. Let us follow a level-wise randomization strategy on the d-level recursion tree. In precise terms, for each level $\ell = 1, 2, \ldots, d$ of the recursion tree, draw three independent uniform random permutations $\sigma_\ell, \tau_\ell, \upsilon_\ell$ of $\{1, 2\}$. At level ℓ of the recursion tree, implement the recursive step with a version of (3) resulting from the substitutions

$$\begin{array}{lll} A_{11} \mapsto A_{1^{\sigma_\ell}1^{\tau_\ell}} & B_{11} \mapsto B_{1^{\tau_\ell}1^{\upsilon_\ell}} & C_{11} \mapsto C_{1^{\sigma_\ell}1^{\upsilon_\ell}} \\ A_{12} \mapsto A_{1^{\sigma_\ell}2^{\tau_\ell}} & B_{12} \mapsto B_{1^{\tau_\ell}2^{\upsilon_\ell}} & C_{12} \mapsto C_{1^{\sigma_\ell}2^{\upsilon_\ell}} \\ A_{21} \mapsto A_{2^{\sigma_\ell}1^{\tau_\ell}} & B_{21} \mapsto B_{2^{\tau_\ell}1^{\upsilon_\ell}} & C_{21} \mapsto C_{2^{\sigma_\ell}1^{\upsilon_\ell}} \\ A_{22} \mapsto A_{2^{\sigma_\ell}2^{\tau_\ell}} & B_{22} \mapsto B_{2^{\tau_\ell}2^{\upsilon_\ell}} & C_{22} \mapsto C_{2^{\sigma_\ell}2^{\upsilon_\ell}} \end{array} \quad . \tag{5}$$

This randomization causes any fixed product among the 8^d products to be present in the broken output uniformly at random, with probability $(\frac{7}{8})^d$.

Repetition enables Boolean matrix multiplication. By making independent repeats of the randomized algorithm, it thus takes, in expectation, $\left(\frac{8}{7}\right)^d \cdot O(6^d) =$

[1] Addition-optimal in the standard basis, the number of additions decreases to 12 when employing the recent alternative-basis framework of Karstadt and Schwartz [49].

[2] Here we have taken the liberty to rephrase the Strassen–Winograd algorithm somewhat to better highlight our instrumentation of the algorithm to follow.

$O\big(\big(6+\frac{6}{7}\big)^d\big)$ operations to witness any fixed product among the 8^d products as present in at least one repeat. With some further randomization of the operands at the start of each repeat, $\lceil 3d\left(\frac{8}{7}\right)^d\rceil$ repeats, each running in $O(6^d)$ operations, suffice to correctly multiply two Boolean matrices, viewed as bit matrices consisting of integers modulo 2 for purposes of the algorithm, with high probability.[3] In particular, we obtain a randomized algorithm for $n \times n$ Boolean matrix multiplication that runs in $O(n^{\log_2(6+\frac{6}{7})} \log n)$ operations, which is faster than the $O(n^{\log_2 7})$ operations for the Strassen–Winograd algorithm. Furthermore, each individual repeat runs in $O(n^{\log_2 6})$ operations.

Opportunistic access to output. Each repeat gives *opportunistic* access to the final result: any nonzero entry in the output of a repeat will be a 1-entry in the final Boolean product, which gives the possibility for an early termination in applications that seek 1-entries in the result. One such application is detecting whether a given graph contains at least one triangle. In fact, a single repeat of the randomized algorithm, executed over the ring of integers, can also be turned into an unbiased estimator for the number of triangles in a graph.

1.2 Tensors and Representations of Tensors.

Let us now switch to the language of tensors and place our present work into context before stating our results. Formal definitions can be found in §2.

It is well known [14, 21] that the study of algebraic complexity of matrix multiplication over a field $\mathbb{F}$ reduces to study of tensors $\langle s,t,u\rangle$ that represent matrix multiplication as a bilinear map $\mathbb{F}^{s\times t} \times \mathbb{F}^{t\times u} \to \mathbb{F}^{s\times u}$. In particular, the exponent $\omega = \omega(\mathbb{F})$ of matrix multiplication over $\mathbb{F}$ can be characterized asymptotically (e.g. [21, Proposition 15.1]) via the tensor rank by

$$\omega = \inf\{\tau : \operatorname{rk}\langle s,s,s\rangle = O(s^\tau)\}\ .$$

Thus, the design of asymptotically fast matrix multiplication algorithms reduces to study of tensor rank of matrix multiplication tensors. Furthermore, a nontrivial upper bound for the rank $\operatorname{rk}\langle s,t,u\rangle$ of any fixed tensor $\langle s,t,u\rangle$ gives both

(i) an upper bound $\omega \le \log_{(stu)^{1/3}} \operatorname{rk}\langle s,t,u\rangle$, and

(ii) an algorithm that multiplies $n \times n$ matrices in $O\big(n^{\log_{(stu)^{1/3}} \operatorname{rk}\langle s,t,u\rangle}\big)$ operations in $\mathbb{F}$.

For example, Strassen's 2×2 algorithm [69] shows that $\operatorname{rk}\langle 2,2,2\rangle \le 7$ and thus $\omega \le \log_2 7$. (In fact, $\operatorname{rk}\langle 2,2,2\rangle = 7$ [41, 75].)

One obstacle to the study of tensors is that many computational problems that are tractable for matrix inputs become NP-hard when the inputs are higher-order tensors [40]. This is the case for tensor rank in particular [39].

Since the direct study of large matrix multiplication tensors $\langle s,t,u\rangle$ appears hard (cf. §1.5), many of the fundamental advances in the complexity of matrix multiplication in the 50 years since Strassen's [69] initial breakthrough have originated from

(a) novel *representations* of tensors, and

(b) new constructions for converting low-rank decompositions in a representation into low-rank decompositions of matrix multiplication tensors.

The first asymptotic improvement over Strassen's algorithm was obtained by Pan [61] through a trilinear aggregation–cancellation technique. Bini [13] introduced the technique of representing a tensor with arbitrary-precision approximations of the tensor, leading to the geometric [53] notion of the *border rank* of a tensor. Schönhage [66] showed how to construct low-rank decompositions of matrix multiplication tensors from low-rank and low-border-rank decompositions of partial and disjoint matrix multiplication tensors. Strassen's *laser method* [70] introduced a powerful construction for low-rank decompositions that underlies the current asymptotically fastest designs for matrix multiplication, due to Coppersmith and Winograd [29], Davie and Stothers [33], Vassilevska Williams [72], and Le Gall [56]. In the process of seeking improved upper bounds on ω via embeddings of matrix multiplication into other algebras [26, 27, 28], Cohn and Umans [28] introduced a new representation for a tensor using tensors that have the same *support* of nonzero entries as the tensor being represented. The resulting notion of *support rank* and the associated *support rank exponent* ω_{s} with $2 \le \omega_{\mathrm{s}} \le \omega$ and $\omega \le \frac{3}{2}\omega_{\mathrm{s}} - 1$ can be used to characterize nondeterministic quantum communication complexity [20] and to establish that $n \times n$ Boolean matrix multiplication can be performend in $O(n^{\omega_{\mathrm{s}}+o(1)})$ operations [28].[4] Recently, Christandl and Zuiddam [25] develop a *tensor surgery* technique for proving upper bounds on tensor ranks. Pan [62] gives a recent comprehensive survey of fast matrix multiplication.

In this paper our motivation is to extend this body of work by studying a probabilistic notion of tensor

[3] A detailed exposition and analysis can be found in §1.4.

[4] Indeed, observe that the best known inequalities $2 \le \omega_{\mathrm{s}} \le \omega < 2.3728639$ and $\omega \le \frac{3}{2}\omega_{\mathrm{s}} - 1$ due to Le Gall [56] and Cohn and Umans [28], respectively, leave open the possibility that $\omega_{\mathrm{s}} < \omega$ and thus the possibility that matrix multiplication over $\mathbb{F}$ has strictly higher asymptotic complexity than Boolean matrix multiplication.

rank, which enables us to gain algorithmically over previous approaches based on deterministic notions of tensor rank already for small tensors, such as $\langle 2,2,2\rangle$.

1.3 Our Results—Coarse-Grained View. We start with the key definitions and a coarse-grained view to our algorithmic results.

Probabilistic tensors. We work over a fixed but arbitrary field $\mathbb{F}$ of scalars, and our tensors are defined with respect to arbitrary but fixed bases for the constituent vector spaces so that the entries of a tensor and its support and shape are well-defined notions (cf. §2 for our detailed conventions).

Definition 1.1. (Probabilistic tensor) *A* probabilistic tensor *is a probability distribution over a space of tensors.*

What makes probabilistic tensors useful from an algorithmic point of view is that they enable one to represent a target tensor probabilistically using tensors whose rank is, in expectation, strictly less than the rank of the target tensor. This lesser rank translates to reduced algebraic complexity and thus faster algorithms in applications, assuming the probabilistic representation retains the application-essential properties of the target tensor with sufficient probability.[5] The next definition is motivated by the subsequent applications in Boolean matrix multiplication and in triangle detection.

Let T be a tensor and let $\tilde{T}$ be a probabilistic tensor over the same space of tensors that contains T. We say that $\tilde{T}$ *supports* T *entrywise with probability at least* p if for every entry in the tensor $\tilde{T}$ it holds that the entry either is zero or is equal to the corresponding entry of T, and equality occurs with probability at least p.

Definition 1.2. (Probabilistic tensor rank) *The* probabilistic rank *of a nonzero tensor* T *is the minimum of* $\mathbb{E}_{S\in\tilde{T}}[\operatorname{rk} S]p^{-1}$ *over all probabilistic tensors* $\tilde{T}$ *that support* T *entrywise with probability at least* $p>0$. *The probabilistic rank of a zero tensor is* 0. *We write* $\widetilde{\operatorname{rk}}\,T$ *for the probabilistic rank of* T.

We observe that it is not immediate that the minimum exists in Definition 1.2, but we will see that this is the case through a characterization by linear programming (cf. §3.2). The *probabilistic border rank* is defined by replacing the tensor rank $\operatorname{rk} S$ in Definition 1.2 with the border rank $\underline{\operatorname{rk}}\,S$. We write $\widetilde{\underline{\operatorname{rk}}}\,T$ for the probabilistic border rank of T.

[5] This general situation of reduced algebraic complexity obtained through randomization arises frequently in algorithm design, cf. §1.5.

Since the border rank and rank satisfy $\underline{\operatorname{rk}}\,S \le \operatorname{rk} S$, it is immediate that $\widetilde{\underline{\operatorname{rk}}}\,T \le \widetilde{\operatorname{rk}}\,T \le \operatorname{rk} T$. Furthermore, we will show that the probabilistic rank and probabilistic border rank retain some natural properties of a rank function, such as submultiplicativity $\widetilde{\operatorname{rk}}\,T\otimes T' \le \widetilde{\operatorname{rk}}\,T\cdot\widetilde{\operatorname{rk}}\,T'$ under Kronecker products of tensors.

Boolean matrix multiplication—the coarse-grained view. The following theorem presents a coarse-grained view to the algorithm designs enabled by probabilistic rank. A similar theorem holds for probabilistic border rank, but we omit a detailed study of border rank from the present conference version of this paper.

Theorem 1.1. (Boolean matrix multiplication) *For all integers* $s,t,u\ge 2$ *and any constant* $\epsilon>0$, *there exists a randomized algorithm that with high probability multiplies two given* $n\times n$ *matrices over the Boolean algebra in*

$$O\big(n^{(\log_{(stu)^{1/3}}\widetilde{\operatorname{rk}}\,\langle s,t,u\rangle)+\epsilon}\big) \tag{6}$$

operations in the field of scalars for $\langle s,t,u\rangle$.

Here the upper bound (6) in general can hide an impractically large constant that depends on the constants s,t,u,ϵ and the field of scalars $\mathbb{F}$ for $\langle s,t,u\rangle$. Furthermore, the upper bound can be turned into running time by accounting for the time cost of arithmetic in $\mathbb{F}$ in the assumed model of computation. Our subsequent more fine-grained analysis will bound these constants for specific small s,t,u and will employ the binary field $\mathbb{F}_2$ (cf. §1.4).

Classical tensor-based fast matrix multiplication algorithms enable a version of Theorem 1.1 with the probabilistic rank replaced by a corresponding classical (deterministic) rank. Furthermore, the work of Cohn and Umans [28, Theorem 3.7] on support rank essentially contains a similar theorem with the probabilistic rank replaced by corresponding (deterministic) support rank.

When applied to a fixed base tensor $\langle s,t,u\rangle$, the present probabilistic tensor framework thus yields asymptotically faster algorithms for Boolean matrix multiplication *precisely when the probabilistic rank is strictly less than the deterministic rank.*

Probabilistic rank of matrix multiplication. We proceed to study the probabilistic rank of matrix multiplication tensors and find that probabilistic rank gains on all the deterministic ranks already in the smallest nontrivial case $\langle 2,2,2\rangle$.

Theorem 1.2. (Bounds on probabilistic rank) *Over any ring, it holds that*

$$\widetilde{\operatorname{rk}}\,\langle 2,2,2\rangle \le 6+\frac{6}{7} \quad \textit{and} \quad \widetilde{\underline{\operatorname{rk}}}\,\langle 2,2,2\rangle \le 6+\frac{2}{3}\,.$$

The upper bounds in Theorem 1.2 should be contrasted with known results on tensor rank, due to Hopcroft and Kerr [41], Winograd [75], on border rank, due to Landsberg [52], and on border support rank, due to Bläser, Christandl, and Zuiddam [15]:

$$\operatorname{rk}\langle 2,2,2\rangle = 7\,,\ \underline{\operatorname{rk}}\,\langle 2,2,2\rangle = 7\,,\ \text{and}\ \underline{\operatorname{rk}}_{\mathrm{s}}\,\langle 2,2,2\rangle = 7\,.$$

In particular, none of these notions of rank is able to witness further structure in the tensor $\langle 2,2,2\rangle$ beyond Strassen's original discovery. Probabilistic tensors and associated probabilistic notions of tensor rank thus reinforce these existing notions so that further algorithmically serendipitous structure and faster algorithms can be uncovered.

Our key technical tool towards bounding the probabilistic rank of matrix multiplication is the following lemma that enables us to study tensors with support-transitive symmetry using individual subtensors. For a tensor T, we write $\operatorname{wt} T$ for the number of nonzero entries in T, or the *weight* of T. We state the lemma for probabilistic rank and rank, but a similar lemma holds for probabilistic border rank and border rank.

LEMMA 1.1. (SUPPORT-TRANSITIVE TENSORS) *Let T be a tensor that admits a group of permutation automorphisms that acts transitively on the support of T. Then, for any subtensor S of T we have the inequality*

$$\operatorname{wt} S \,\widetilde{\operatorname{rk}}\, T \leq \operatorname{wt} T \operatorname{rk} S\,. \tag{7}$$

Moreover, equality holds for at least one subtensor.

Together with the elementary inequality

$$\operatorname{rk} T \leq \operatorname{rk} S + \operatorname{wt} T - \operatorname{wt} S \tag{8}$$

we thus obtain two-sided control on the probabilistic rank and observe that the probabilistic rank can be strictly less than the rank only if $\operatorname{rk} T < \operatorname{wt} T$.

Since matrix multiplication tensors are support-transitive, Lemma 1.1 enables us to localize the study of probabilistic rank of matrix multiplication to the study of rank-weight ratios of individual subtensors S. In particular, this enables us to prove Theorem 1.2 using the broken Strassen–Winograd decomposition (3) and a decomposition of Bini [13] for the rank and border rank, respectively. Lemma 1.1 also enables numerical study of probabilistic rank of small tensors, by reducing to numerical study of rank of subtensors of the matrix multiplication tensor (cf. §1.5); such a numerical study will not be conducted in the present paper, however.

Associated with any probabilistic notion is the question whether it can be efficiently derandomized. We show that probabilistic rank admits at least a partial asymptotic derandomization using a variant of Adleman's argument [3]:

THEOREM 1.3. *Over the complex field, it holds that*

$$\omega_{\mathrm{s}} \leq \inf\bigl\{\tau : \widetilde{\operatorname{rk}}\,\langle t,t,t\rangle = O(t^{\tau})\bigr\}\,. \tag{9}$$

Crucially, however, we have not been able to establish similar derandomization over *finite* fields, and the binary field in particular.

This leaves open the possibility that probabilistic rank may be asymptotically more powerful than deterministic rank, in particular for the applications of Boolean matrix multiplication and triangle detection, where the binary field $\mathbb{F}_2$ suffices. With the binary field in mind, let us now move to a more fine-grained setting.

1.4 Our Results—Fine-Grained View. From a practical standpoint, the coarse-grained view presented in the previous section is in many ways unsatisfactory. This is in particular because the elementary cubic algorithm for Boolean matrix multiplication admits extremely efficient implementations both in software and in hardware. In essence, for $n \times n$ Boolean inputs, the elementary cubic algorithm performs exactly $n^3 - n^2$ Boolean disjunctions and n^3 Boolean conjunctions, yielding a total of $2n^3 - n^2$ bit operations, which in a software implementation can be executed using word operations and vectorization up to the lengths supported in the underlying hardware.

Our fine-grained view to probabilistic rank and associated algorithms will proceed by means of a detailed look at the underlying tensor representations, which will be conveniently manipulated via the following definition. Here it should be noted that such a template-based view to the study of matrix multiplication is standard practice, and we merely extend this view to the case of proper subtensors of a matrix multiplication tensor; that is, to the case $w < stu$ in the definition below. In fact, also the use of subtensors is standard at least since the work of Schönhage [66]. Where our present work differs is in the use of randomization and in the somewhat expanded notion of a subtensor. For a more detailed discussion of the latter, cf. §2.5.

DEFINITION 1.3. (TEMPLATE) *A* template *with parameters $\langle s,t,u|m,w\rangle$ is a representation[6] of a weight-w subtensor of the matrix multiplication tensor $\langle s,t,u\rangle$ using at most m multiplications.*

When $w = stu$, we omit the weight w from the notation and write $\langle s,t,u|m\rangle$. Similarly, when $s = t = u$, we may write simply $\langle s|m,w\rangle$.

[6] For what it means to represent a tensor using a given number of multiplications, cf. §2.3 for tensor rank and §2.6 for border rank.

For example, the 2×2 Strassen–Winograd algorithm (1) with its 7 multiplications essentially defines a $\langle 2|7\rangle$-template, and the broken version (3) of the algorithm essentially defines a $\langle 2|6,7\rangle$-template, where the 6 records the number of multiplications in the algorithm, and the 7 records the number of products remaining in (4). For the border rank, a well-known decomposition of Bini [13] gives a $\langle 2|5,6\rangle$-template. These templates exist for all fields. We postpone an exposition to §4.

The next theorem shows that templates give us control over probabilistic rank. Although we state the result as a theorem, the actual work was already done in Lemma 1.1. Indeed, Theorem 1.4 is an immediate corollary of the definition of tensor rank (cf. §2.3), Lemma 1.1, and the fact that matrix multiplication tensors are support-transitive (cf. §2.8). A similar theorem holds for probabilistic border rank.

THEOREM 1.4. *Suppose that there exists a template with parameters $\langle s,t,u|m,w\rangle$. Then,*

$$\widetilde{\mathrm{rk}}\,\langle s,t,u\rangle \leq \frac{mstu}{w}\,.$$

Moreover, equality holds for at least one template.

Theorem 1.2 now follows as a corollary of Theorem 1.4 and the aforementioned templates for $\langle 2,2,2\rangle$. Indeed, the $\langle 2|6,7\rangle$-template gives

$$\widetilde{\mathrm{rk}}\,\langle 2,2,2\rangle \leq \frac{6\cdot 2^3}{7} = 6+\frac{6}{7}\,.$$

The $\langle 2|5,6\rangle$-template for border rank gives

$$\underline{\widetilde{\mathrm{rk}}}\,\langle 2,2,2\rangle \leq \frac{5\cdot 2^3}{6} = 6+\frac{2}{3}\,.$$

While templates control probabilistic rank, the notion of a template in itself does not yet convey precise-enough information about arithmetic complexity to enable a bit-operation-level comparison of our algorithms and the elementary cubic algorithm for Boolean matrix multiplication.

For a yet more fine-grained view, let us move to the level of templates describing sum–product–sum circuits built from a template by preprocessing the input and postprocessing the output of multiplications using fan-in-two addition gates.[7] We stress that such templates are standard practice in the case $w = stu$, which reduces to the design of fine-grained fast matrix multiplication algorithms over $\mathbb{F}$.[8] Our contribution here amounts to the observation that this design framework extends to the case $w < stu$ too, and thus to algorithms relying on low probabilistic rank rather than deterministic rank.

DEFINITION 1.4. (CIRCUIT TEMPLATE) *A* circuit template *with parameters $\langle s,t,u|m,w\rangle^{\bar{a},\bar{b},\bar{c}}_{\underline{a},\underline{b},\underline{c}}$ is a template for sum–product–sum circuits that implement an $\langle s,t,u|m,w\rangle$-template for two selected input shapes among the three shapes $s\times t$, $t\times u$, and $u\times s$, with output given in the remaining third shape. A circuit* instantiated *from the template consists of at most m fan-in-two multiplication gates and at most $\bar{a},\bar{b},\bar{c}$ fan-in-two addition gates for the $s\times t$, $t\times u$, or $u\times s$ inputs, respectively, and at most $\underline{a},\underline{b},\underline{c}$ fan-in-two addition gates for the remaining $s\times t$, $t\times u$, or $u\times s$ output, respectively.*[9]

When $\bar{a}=\bar{b}=\bar{c}$ and $\underline{a}=\underline{b}=\underline{c}$, we write simply $\langle s,t,u|m,w\rangle^{\bar{a}}_{\underline{a}}$. Recalling the introduction, the Strassen–Winograd algorithm essentially defines a $\langle 2|7\rangle^4_7$-circuit template and its broken version defines a $\langle 2|6,7\rangle^4_6$-circuit template. For the detailed templates with the addition gates given explicitly, please consult (33).

A key property of circuit templates is that they can be composed, and composition gives precise analytical control on the arithmetic complexity in terms of the number of fan-in-two gates in the instantiated circuit. The following lemma states a special case for repeated composition of a circuit template with itself.

LEMMA 1.2. (POWERING A CIRCUIT TEMPLATE) *Suppose that there exists a circuit template with parameters $\langle s|m,w\rangle^{\bar{a}}_{\underline{a}}$. Then, for any positive integer d, there exists a circuit template with parameters $\langle s^d|m^d,w^d\rangle^{D\cdot\bar{a}}_{D\cdot\underline{a}}$, where*

$$D = \begin{cases} ds^{2d} & \text{if } m = s^2\,,\\ \frac{m^d-s^{2d}}{m-s^2} & \text{if } m\neq s^2\,. \end{cases} \tag{10}$$

We record an immediate corollary for the $\langle 2|7\rangle^4_7$-circuit template and the $\langle 2|6,7\rangle^4_6$-circuit template.

LEMMA 1.3. (TWO TEMPLATE FAMILIES) *Over any field and for all positive integers d, there exist circuit templates with parameters*

$$\langle 2^d|7^d\rangle^{4(7^d-4^d)/3}_{7(7^d-4^d)/3} \quad \text{and} \quad \langle 2^d|6^d,7^d\rangle^{2(6^d-4^d)}_{3(6^d-4^d)}\,.$$

[7] We adopt the convention that a fan-in-two addition gate allows the multiplication of its two inputs by arbitrary scalar constants. For the binary field $\mathbb{F}_2$, such extended addition gates reduce to addition gates without the constant multilications. Our convention with border rank is to view univariate polynomials with coefficients in $\mathbb{F}$ as scalars, and then transform to a circuit over $\mathbb{F}$ either (i) via evaluation–interpolation over the error degree or (ii) by expanding with subcircuits for polynomial arithmetic. Cf. §4.1.

[8] For example, see Karstadt and Schwartz [49] for the state of the art, as well as an alternative-basis extension of the basic fine-grained theory which we will, however, not pursue here.

[9] For example, a circuit instantiated with inputs of shape $s\times t$ and $t\times u$ will consist of at most m multiplication gates and at most $\bar{a}+\bar{b}+\underline{c}$ addition gates. The circuit will give an output of shape $u\times s$.

Let us now turn to Boolean matrix multiplication and present our key technical result underlying Theorem 1.1. In essence, we use a circuit template over $\mathbb{F}_2$ together with elementary randomization techniques to arrive at a randomized sketch R of the Boolean product of two matrices U and V. Again our contribution is that this approach extends to the case $w < stu$, whereas the case $w = stu$ is well known.

THEOREM 1.5. (OPPORTUNISTIC SKETCHING) *Suppose that there exists a circuit template with parameters $\langle s,t,u|m,w\rangle^{\bar{a},\bar{b},\bar{c}}_{\underline{a},\underline{b},\underline{c}}$ over the binary field $\mathbb{F}_2$. Then, there exists a randomized algorithm that on input $U \in \{0,1\}^{s\times t}$ and $V \in \{0,1\}^{t\times u}$ gives an output $R \in \{0,1\}^{s\times u}$ such that for all $i \in [s]$ and $k \in [u]$ it holds that*

(i) $\bigvee_{j\in[t]} U_{ij} \wedge V_{jk} = 0$ *implies* $\Pr[R_{ik} = 0] = 1$, *and*

(ii) $\bigvee_{j\in[t]} U_{ij} \wedge V_{jk} = 1$ *implies* $\Pr[R_{ik} = 1] \geq \frac{w}{2stu}$.

Moreover, the algorithm makes at most $\bar{a}+\bar{b}+\underline{c}$ additions and at most m multiplications in $\mathbb{F}_2$.

The properties (i) and (ii) in Theorem 1.5 state that, compared with the correct Boolean product, the sketch R never contains an erroneous 1-entry, but erroneous 0-entries can occur. Let us say that the lower bound $\frac{w}{2stu}$ in (ii) is the *witnessing probability* of the sketch. In particular, the witnessing probability is a lower bound on the probability to witness any fixed 1-entry in the correct Boolean product via a single sketch.

The witnessing probability can be amplified from the value p for a single sketch to $1-\epsilon$ for any $\epsilon > 0$ by repeating the sketch with independent randomness at least $r = \lceil p^{-1} \ln \epsilon^{-1} \rceil$ times and taking the disjunction of the sketches obtained.[10] Setting $\epsilon = \delta/(su)$ and using the union bound, we obtain that at least $\lceil p^{-1}(\ln \delta^{-1} + \ln su)\rceil$ repeated sketches recover the correct Boolean product with probability at least $1-\delta$.

A key application of witnessing probability is *triangle detection* in graphs; that is, the problem of deciding whether a given graph contains at least one triangle. Indeed, the vertices i,j,k in a graph with adjacency matrix A form a triangle if and only if $A_{ij} = A_{ik} = A_{jk} = 1$. This triangle can be witnessed from the value $A_{ik} = 1$ and the fact that the entry at position ik of the Boolean product of A with itself is 1. Thus, the witnessing probability is the probability with which the first triangle (if any) will be discovered, in the worst case.

We show that our present framework is practical (as measured by the number of bit operations) for triangle detection, and in fact modestly outperforms both the elementary cubic algorithm and the Strassen–Winograd framework starting from rather small parameter values.

The table below records the number of bit operations $B(2^d)$ and the witnessing probability $p(2^d)$ for a single sketch of $2^d \times 2^d$ Boolean matrix multiplication arising from Theorem 1.5 applied to the templates in Lemma 1.3:

Template	$B(2^d)$	$p(2^d)$
$\mathbb{B}$	$2\cdot 8^d - 4^d$	deterministic
$\langle 2\|7\rangle^4_7$	$6\cdot 7^d - 5\cdot 4^d$	2^{-1}
$\langle 2\|6,7\rangle^4_6$	$8\cdot 6^d - 7\cdot 4^d$	$2^{-1}(7/8)^d$

Let us now write $T(2^d) = B(2^d)\lceil p^{-1}(2^d)\rceil$ for the number of bit operations sufficient for witnessing probability $1-\frac{1}{e} \geq 0.632$. In particular, this is the probability to detect a triangle (if any), in the worst case. The next table shows that the broken Strassen–Winograd algorithm gains on both the Strassen–Winograd algorithm and the elementary cubic algorithm starting from $d \geq 14$:

d	$\frac{B_{\mathbb{B}}(2^d)}{T_{\langle 2\|7\rangle^5_7}(2^d)}$	$\frac{B_{\mathbb{B}}(2^d)}{T_{\langle 2\|6,7\rangle^4_6}(2^d)}$	$\frac{T_{\langle 2\|7\rangle^5_7}(2^d)}{T_{\langle 2\|6,7\rangle^4_6}(2^d)}$
10	0.635	0.563	0.887
11	0.725	0.664	0.916
12	0.828	0.794	0.959
13	0.946	0.881	0.931
14	1.081	1.083	1.001
15	1.235	1.250	1.012
16	1.412	1.469	1.041
17	1.613	1.664	1.032
18	1.844	1.929	1.046
19	2.107	2.275	1.080
20	2.408	2.719	1.129

Here the key finding motivating the fine-grained analysis is not in the magnitude of the gain, which is modest at best, but the overall feasibility of the probabilistic framework to yield algorithm designs that do not hide large constants.[11] Further work and careful tailoring of the templates is needed to arrive at actual practical implementations. For example, the alternative-basis approach of Karstadt and Schwartz [49] can be employed to optimize the constants while maintaining cache-efficiency through randomization generalizing (5). Furthermore, it is possible to optimize the specific subtensor of weight w underlying the template. In particular, different tensors of the same weight and rank

[10] Indeed, we have $(1-p)^r \leq \exp(-pr) \leq \epsilon$.

[11] We refer to Pan's recent survey [62] for a discussion of practically feasible and infeasible designs in the study of matrix multiplication.

can differ in their additive characteristics. Yet further potentially applicable techniques to speed up deterministic fast matrix multiplication are developed by Cenk and Hasan [22] and Bodrato [17].

We also observe that the present framework can be parallelized using the communication-avoiding paradigm [7, 8, 9, 57] that enables parallelization with strong scaling in both shared-memory and distributed-memory architectures; however, the present paper does not pursue this direction. Rather we are content to observe that probabilistic rank enables one to witness algorithmically serendipitous structure beyond existing (deterministic) notions of tensor rank, with possible further potential in practical applications and theoretical work. For example, the probabilistic rank of $\langle 2,2,2\rangle$ is not yet known; Theorem 1.2 only gives upper bounds. We find it exciting that yet larger gains over the deterministic framework are a possibility for $\langle 2,2,2\rangle$ and other small tensors.

1.5 Related Work. Our present setting of probabilistic tensors and probabilistic tensor rank motivated by Boolean matrix multiplication is analogous to a number of applications of randomization and algebraic techniques in algorithm design and complexity, including the *probabilistic polynomials* of Razborov [65] and Smolensky [68] in circuit complexity together with their applications and extensions in fine-grained algorithm design (e.g. [2, 4, 74]). Algorithmic applications of matrix rank, randomization, and derandomization are considered in a number of recent works, including Alman and Williams [5], Bodlaender, Cygan, Kratsch, and Nederlof [16], Cygan, Kratsch, and Nederlof [30], Cygan, Nederlof, Pilipczuk, Pilipczuk, van Rooij, and Wojtaszczyk [31], and Fomin, Lokshtanov, Panolan, and Saurabh [36]. Approximate randomized matrix multiplication techniques are developed in Drineas, Kannan, and Mahoney [34], Pan, Luan, Svadlenka, and Zhao [63], and Pagh [59].

Although computing tensor ranks is hard [39], the tensor rank of small matrix multiplication tensors can be studied numerically using systems of cubic polynomial equations, cf. Brent [18]. Recent work in this direction includes Smirnov [67] and Benson and Ballard [12]. Similarly, techniques from algebraic geometry can be used to study border rank (cf. Landsberg [53]). Recent work in this direction in the context of small tensors includes Chiantini, Ikenmeyer, Landsberg, and Ottaviani [24], Ballard, Ikenmeyer, Landsberg, and Ryder [10], Landsberg and Michałek [54], and Landsberg and Ryder [55].

2 Preliminaries.

This section reviews terminology and notational conventions used in this paper. Throughout this paper we work over a fixed but arbitrary field $\mathbb{F}$ unless indicated otherwise, with the understanding that the relevant notions studied in general tacitly depend on the field.

For a nonnegative integer k, we write $[k]$ for the set $\{1,2,\ldots,k\}$. For a logical proposition P, we use Iverson's bracket notation $[\![P]\!]$ to indicate a 1 if P is true and a 0 if P is false. We write $\log_b x = \frac{\log x}{\log b}$ for the base b logarithm of x. If the base is omitted, $b = 2$ is assumed. We write $\ln x$ for the logarithm of x in the natural base $\sum_{i=0}^{\infty} \frac{1}{i!}$.

2.1 Tensors. Tensor admit many natural representations (e.g. [28, 50, 53]). In this paper we adopt the convention of viewing tensors in a basis-dependent representation either as arrays of a particular shape with entries in $\mathbb{F}$, or as multilinear forms over formal variables, where the formal variables are partitioned into groups, one group for each mode of the tensor.[12]

For example, let us represent an $m \times n$ matrix $A \in \mathbb{F}^{m\times n}$ with entries $A_{ij} \in \mathbb{F}$ for $i \in [m]$ and $j \in [n]$ as a tensor of order two in the multilinear representation. Introduce two groups of formal variables, $x_1, x_2, \ldots, x_m$ and $y_1, y_2, \ldots, y_n$, corresponding to the rows and columns of the matrix. Formally, we can now represent the matrix as the multilinear form $A = \sum_{i\in[m],\, j\in[n]} A_{ij} x_i y_j$ in the polynomial ring $\mathbb{F}[x_1, x_2, \ldots, x_m, y_1, y_2, \ldots, y_n]$.

Similarly, we can represent tensors of order three using three groups of formal variables x_i, y_j, z_k with $i \in I$, $j \in J$, and $k \in K$ for three nonempty finite sets I, J, K of indices. A tensor $T \in F^{I\times J\times K}$ of *shape* $|I| \times |J| \times |K|$ with entries $T_{ijk} \in \mathbb{F}$ can now represented by the trilinear form $T = \sum_{i\in I,\, j\in J,\, k\in K} T_{ijk} x_i y_j z_k$ in the polynomial ring $\mathbb{F}[x,y,z] = \mathbb{F}[x_i, y_j, z_k : i \in I, j \in J, k \in K]$. Unless indicated otherwise, notationally we adopt the convention that the symbols x, y, z are reserved for formal variables, and the symbol x alone without a subscript stands for the entire group of variables x_i for $i \in I$. We will also omit the index sets I, J, K and tacitly assume that a sum over an index ranges over the entire relevant index set. For example, we write $\sum_{i,i'}$ to indicate $\sum_{i\in I, i'\in I'}$ for two index sets I and I'.

[12] Compared with a basis-free approach (e.g. [53]), these representations tacitly depend on fixed but arbitrary bases chosen for each component vector space. This representation is not the most general possible but natural for purposes of computation where bases must be fixed for computation in terms of scalar operations to take place.

Two tensors viewed as multilinear forms are *isomorphic* if one can be obtained from the other by an invertible linear change of variables in each group of variables x, y, z. We write $T \cong T'$ to indicate that T and T' are isomorphic. Two tensors are *permutation isomorphic* if the change of variables in each group can be represented by a permutation matrix. In essence, permutation isomorphism is isomorphism by permutation of the variables in each group. Such a tuple of permutations, one permutation for each group of variables, is a *permutation isomorphism*. A permutation isomorphism that maps a tensor to itself is a *permutation automorphism*.

Let us next review standard operations on tensors. Let $T = \sum_{i,j,k} t_{ijk} x_i y_j z_k$ and $T' = \sum_{i',j',k'} t'_{i'j'k'} x'_{i'} y'_{j'} z'_{k'}$ be two tensors of order three. The *Kronecker product* of T and T' is the tensor $T'' = T \otimes T'$ defined by

$$T'' = \sum_{i,i',j,j',k,k'} t_{ijk} t'_{i'j'k'} x''_{ii'} y''_{jj'} z''_{kk'} \,. \tag{11}$$

In particular, if T has shape $\ell \times m \times n$ and T' has shape $\ell' \times m' \times n'$, then $T \otimes T'$ has shape $\ell\ell' \times mm' \times nn'$.

2.2 Matrix Multiplication as a Tensor. For positive integers s, t, u, the map $\mathbb{F}^{s\times t} \times \mathbb{F}^{t\times u} \to \mathbb{F}^{s\times u}$ that multiplies a matrix of shape $s \times t$ with a matrix of shape $t \times u$ is bilinear. We adopt the following representation for this map using a tensor of order three.[13] Let us write $\langle s, t, u\rangle$ for the tensor of shape $st \times tu \times us$ defined by

$$\sum_{\substack{i_1\in[s],\, i_2\in[s],\\ j_1\in[t],\, j_2\in[t],\\ k_1\in[u],\, k_2\in[u]}} [\![i_1 = i_2]\!][\![j_1 = j_2]\!][\![k_1 = k_2]\!] x_{i_1 j_2} y_{j_1 k_2} z_{k_1 i_2} \,. \tag{12}$$

For example, the tensor for 2×2 multiplication is

$$\begin{aligned} \langle 2,2,2\rangle = {} & x_{11}y_{11}z_{11} + x_{12}y_{21}z_{11} + x_{11}y_{12}z_{21} + \\ & x_{12}y_{22}z_{21} + x_{11}y_{11}z_{12} + x_{12}y_{21}z_{12} + \\ & x_{21}y_{12}z_{22} + x_{22}y_{22}z_{22} \,. \end{aligned} \tag{13}$$

The family of matrix multiplication tensors is closed under taking of Kronecker products: for all positive integers s, s', t, t', u, u' it holds that

$$\langle s, t, u\rangle \otimes \langle s', t', u'\rangle \cong \langle ss', tt', uu'\rangle \,. \tag{14}$$

Furthermore, the isomorphism in (14) is a permutation isomorphism.

[13] Indeed, whereas a linear map can be represented by a matrix (a tensor of order two), a bilinear map can be represented by a tensor of order three.

2.3 Tensor Rank. Let $T = \sum_{i,j,k} t_{ijk} x_i y_j z_k$ be a tensor of order three. We say that T admits a representation using m *multiplications* if there exist linear forms $\alpha_\ell(x) \in \mathbb{F}[x]$, $\beta_\ell(y) \in \mathbb{F}[y]$, and $\gamma_\ell(z) \in \mathbb{F}[z]$ for $\ell = 1, 2, \ldots, m$ such that

$$T = \sum_{\ell=1}^{m} \alpha_\ell(x)\beta_\ell(y)\gamma_\ell(z) \,. \tag{15}$$

The *rank* of T is the least m such that T admits a representation (15) using m multiplications. We write $\operatorname{rk} T$ for the rank of T. Tensor rank is isomorphism invariant.

It is immediate from (11) and (15) that tensor rank is *submultiplicative* with respect to Kronecker products. That is, for all tensors T and T' of the same order, we have $\operatorname{rk} T \otimes T' \le \operatorname{rk} T \cdot \operatorname{rk} T'$.

To illustrate the connection between tensor rank of matrix multiplication tensors and recursive matrix multiplication algorithms, let us revisit the example in §1.1. From the Strassen–Winograd algorithm (1) we can recover a representation of $\langle 2,2,2\rangle$ using 7 multiplications, as follows. First, substituting $A_{ij} \mapsto x_{ij}$, $B_{ij} \mapsto y_{ij}$, $T_\ell \mapsto \alpha_\ell(x)$ and $S_\ell \mapsto \beta_\ell(y)$ in (1) yields the linear forms

$$\begin{array}{ll} \alpha_1(x) = x_{21} + x_{22} & \beta_1(y) = y_{21} + y_{22} \\ \alpha_2(x) = x_{12} & \beta_2(y) = y_{21} \\ \alpha_3(x) = x_{12} + x_{22} & \beta_3(y) = y_{12} + y_{22} \\ \alpha_4(x) = x_{21} + \alpha_3(x) & \beta_4(y) = y_{21} - \beta_3(y) \\ \alpha_5(x) = x_{11} + \alpha_4(x) & \beta_5(y) = y_{12} \\ \alpha_6(x) = x_{21} & \beta_6(y) = y_{11} - \beta_4(y) \\ \alpha_*(x) = x_{11} & \beta_*(y) = y_{11} \end{array} \,. \tag{16}$$

Second, substituting $C_{ij} \mapsto z_{ji}$ (observe that we intentionally transpose the indices $ij \mapsto ji$ here[14]) and $Q_\ell \mapsto \gamma_\ell(z)$ yields, after simplification, the linear forms

$$\begin{array}{l} z_{11} = \gamma_2(z) + \gamma_*(z) \\ z_{21} = \gamma_5(z) - \gamma_2(z) + \gamma_4(z) - \gamma_1(z) \\ z_{12} = \gamma_6(z) - \gamma_2(z) + \gamma_4(z) + \gamma_3(z) \\ z_{22} = \gamma_1(z) + \gamma_2(z) - \gamma_4(z) - \gamma_3(z) \end{array} \,. \tag{17}$$

Transposing[15] the forms in (17), we obtain

$$\begin{array}{ll} \gamma_1(z) = z_{22} - z_{21} & \gamma_5(z) = z_{21} \\ \gamma_2(z) = z_{11} - z_{21} - z_{12} + z_{22} & \gamma_6(z) = z_{12} \\ \gamma_3(z) = z_{12} - z_{22} & \gamma_*(z) = z_{11} \\ \gamma_4(z) = z_{21} + z_{12} - z_{22} & \end{array} \,. \tag{18}$$

We can now verify that we can realize $\langle 2,2,2\rangle$ using 7 multiplications by substituting (16) and (18) into (15).

[14] This is to obtain the rotational symmetry $x_{i_1 j_2} y_{j_1 k_2} z_{k_1 i_2}$ among the indices in (12).

[15] Represent (17) as the matrix $\left[\begin{smallmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & 0 & 1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & -1 & -1 & 0 & 0 & 0 \end{smallmatrix}\right]$ and transpose the matrix.

This procedure can also be reversed; that is, from (16) and (18) we can mechanically obtain the algorithm (1) by following the procedure in reverse.[16]

Following the same procedure, we observe that the broken Strassen–Winograd algorithm (3) in §1.1 realizes the tensor

$$x_{12}y_{21}z_{11} + x_{11}y_{12}z_{21} + x_{12}y_{22}z_{21} + x_{11}y_{11}z_{12} + x_{12}y_{21}z_{12} + x_{21}y_{12}z_{22} + x_{22}y_{22}z_{22} \tag{19}$$

using the first 6 multiplications from (16) and (18). In particular, observe that the tensor (19) agrees with (13) except for the missing monomial $x_{11}y_{11}z_{11}$, which corresponds to the missing product $A_{11}B_{11}$ in (4).

2.4 Support, Weight, and Support Rank. Let $T = \sum_{i,j,k} t_{ijk} x_i y_j z_k$ be a tensor of order three. The *support* of T is the set of index-tuples of T that contain a nonzero value. In notation, we write $\operatorname{supp} T = \{(i,j,k) \in I \times J \times K : t_{ijk} \neq 0\}$ for the support of T. The *weight* of a tensor is the size of its support. Equivalently, viewing T as a multivariate polynomial, the weight of T is the number of monomials with a nonzero coefficient in T. We write $\operatorname{wt} T = |\operatorname{supp} T|$ for the weight of T. For example, from (12) we have that $\operatorname{wt} \langle s,t,u \rangle = stu$ with

$$\operatorname{supp} \langle s,t,u \rangle = \{(ij, jk, ki) : i \in [s], j \in [t], k \in [u]\}. \tag{20}$$

Cohn and Umans [28] introduced the following generalization of tensor rank. The *support rank* of T is the minimum rank of a tensor T_{s} for which $\operatorname{supp} T_{\mathrm{s}} = \operatorname{supp} T$. We write $\operatorname{rk}_{\mathrm{s}} T$ for the support rank of T. It is immediate that $\operatorname{rk}_{\mathrm{s}} T \leq \operatorname{rk} T$. Furthermore, support rank is easily verified to be submultiplicative but not isomorphism invariant since an isomorphism may alter the support of a tensor. Support rank is invariant under permutation isomorphism.

2.5 Subtensors. Let T be a tensor represented as a multilinear form. We say that a tensor S is a *subtensor* of T if S can be obtained from T by deleting zero or more monomials. Or what is the same, viewing S and T as arrays with the same shape and indexing, S is a subtensor of T if we can obtain S from T by setting zero or more entries in the support of T to zero.

We write $\operatorname{sub} T$ for the set of all subtensors of T. We have $|\operatorname{sub} T| = 2^{\operatorname{wt} T}$. The set of subtensors is clearly basis-dependent and thus not isomorphism invariant.

The present definition of a subtensors is also somewhat more general than what is considered in works that typically address partial tensors associated with matrix multiplication. For example, Schönhage [66] studies subtensors of $\langle s,t,u \rangle$ that are obtained by setting entire *fibers* [50] of entries across the tensor to zero, whereas our present definition allows us to set any entries to zero independently of each other. A similar difference occurs with the study of restrictions and degenerations [70]. In essence, under our present definition $\langle s,t,u \rangle$ has 2^{stu} distinct subtensors, whereas if one works along fibers, the number of subtensors is at most $2^{st+tu+us}$.

2.6 Border Rank. Recall that we can view tensors of order three as trilinear forms $T \in \mathbb{F}[x,y,z]$ with the property that every monomial of T contains exactly one variable from each of the three groups of variables x, y, and z. Let us introduce a new indeterminate ϵ and write $\mathbb{F}_\epsilon = \mathbb{F}[\epsilon]$ for the univariate polynomial ring with indeterminate ϵ and coefficients in $\mathbb{F}$.

Let us now view the tensor T as an element of the polynomial ring $\mathbb{F}_\epsilon[x,y,z]$. We say that T admits a representation using m *multiplications* if there exist (i) a nonnegative integer h, (ii) linear forms $\alpha_\ell(x) \in \mathbb{F}_\epsilon[x]$, $\beta_\ell(y) \in \mathbb{F}_\epsilon[y]$, and $\gamma_\ell(z) \in \mathbb{F}_\epsilon[z]$ for $\ell = 1, 2, \ldots, m$, and (iii) tensors $E^{(1)}, E^{(2)}, \ldots, E^{(e)} \in \mathbb{F}[x,y,z]$ such that

$$\sum_{\ell=1}^{m} \alpha_\ell(x)\beta_\ell(y)\gamma_\ell(z) = \epsilon^h T + \sum_{u=1}^{e} \epsilon^{h+u} E^{(u)} \tag{21}$$

We say the representation (21) has *order* h and that $\epsilon^{h+1}E_1 + \epsilon^{h+2}E_2 + \ldots + \epsilon^{h+e}E_e$ is the *error* associated with the representation. The *error degree* of the representation is e. By truncating the forms as appropriate, without loss of generality we have $e \leq 2h$.

The *border rank* of T is the least m such that T admits a representation (21) using m multiplications. We write $\underline{\operatorname{rk}}\, T$ for the border rank of T. Border rank is isomorphism invariant. Since representations of degree zero reduce to tensor rank, we have $\underline{\operatorname{rk}}\, T \leq \operatorname{rk} T$.

For example, the following linear forms due to Bini [13] give a representation for the weight-6 tensor

$$x_{11}y_{11}z_{11} + x_{12}y_{21}z_{11} + x_{21}y_{11}z_{12} + x_{11}y_{12}z_{21} + x_{12}y_{22}z_{21} + x_{21}y_{12}z_{22} \tag{22}$$

using 5 multiplications:

$$\begin{aligned}
\alpha_1(x) &= \epsilon x_{11} + x_{12} & \beta_1(y) &= y_{12} + \epsilon y_{22} & \gamma_1(z) &= z_{21} \\
\alpha_2(x) &= \epsilon x_{11} + x_{21} & \beta_2(y) &= y_{11} & \gamma_2(z) &= z_{11} + \epsilon z_{12} \\
\alpha_3(x) &= -x_{12} & \beta_3(y) &= y_{12} & \gamma_3(z) &= z_{11} + z_{21} + \epsilon z_{22}\,. \\
\alpha_4(x) &= -x_{21} & \beta_4(y) &= y_{11} + y_{12} + \epsilon y_{21} & \gamma_4(z) &= z_{11} \\
\alpha_5(x) &= x_{12} + x_{21} & \beta_5(y) &= y_{12} + \epsilon y_{21} & \gamma_5(z) &= z_{11} + \epsilon z_{22}
\end{aligned} \tag{23}$$

[16] Optimizing the number of additions in the resulting algorithm, however, is in general a computationally nontrivial task even over the binary field $\mathbb{F}_2$ (see e.g. [46, 49]). For the Strassen–Winograd algorithm, the optimized template can be found in (33).

This representation has order 1 and error degree 2. We also observe that (22) is a subtensor of $\langle 2,2,2\rangle$.

2.7 Groups and Group Actions. Let G be a group and let G act on a set Φ. For $\phi \in \Phi$ and $g \in G$, we write $\phi^g \in \Phi$ for the image of ϕ under the action of g. We write $\phi^G = \{\phi^g : g \in G\}$ for the *orbit* of ϕ under the action of G. We say that G acts *transitively* on Φ if there exists a $\phi \in \Phi$ with $\Phi = \phi^G$. For a finite nonempty set Φ, let us write $\mathrm{Sym}(\Phi)$ for the group of all permutations of Φ with composition of permutations as mappings as the group operation. We say that $\mathrm{Sym}(\Phi)$ is the *symmetric group* on Φ.

Let I, J, and K be finite nonempty index sets and suppose that G acts on each of I, J, K. This induces an action on the set of all tensors in $\mathbb{F}[x,y,z] = \mathbb{F}[x_i, y_j, z_k : i \in I, j \in J, k \in K]$ by extending the monomial-wise action given by $(x_i y_j z_k)^g = x_{i^g} y_{j^g} z_{k^g}$ linearly to sums of monomials.

2.8 Support-Transitivity of Matrix Multiplication. For positive integers s,t,u, suppose that the index sets I, J, and K have the product structure $I = [s]\times[t]$, $J = [t]\times[u]$, and $K = [u]\times[s]$. We let the direct product group $G_{stu} = \mathrm{Sym}([s]) \times \mathrm{Sym}([t]) \times \mathrm{Sym}([u])$ act on $I \times J \times K$ so that $g = (\sigma, \tau, \upsilon)$ acts on a tuple of indices $(i_1 j_2, j_1 k_2, k_1 i_2) \in I \times J \times K$ by

$$(i_1 j_2, j_1 k_2, k_1 i_2)^g = (i_1^\sigma j_2^\tau, j_1^\tau k_2^\upsilon, k_1^\upsilon i_2^\sigma)\,. \tag{24}$$

When extended to tensors of shape $st \times tu \times us$, this action has the property that it fixes the matrix multiplication tensor $\langle s,t,u\rangle$. That is, for all $g \in G_{stu}$ we have $\langle s,t,u\rangle^g = \langle s,t,u\rangle$. Thus, G_{stu} defines a group of permutation automorphisms of $\langle s,t,u\rangle$. Comparing (20) and (24), we also observe that G_{stu} acts transitively on $\operatorname{supp}\langle s,t,u\rangle$. That is, $\langle s,t,u\rangle$ is a support-transitive tensor. Furthermore, the set of subtensors $\operatorname{sub}\langle s,t,u\rangle$ is a union of orbits of the action of G_{stu} on $\mathbb{F}[x,y,z]$. This will be a key property in understanding the probabilistic rank of matrix multiplication tensors.

3 Probabilistic Rank.

This section develops our key tools for working with the probabilitistic rank of a tensor (Definition 1.2). We start by at least partially justifying the term "rank" by showing that probabilistic rank is submultiplicative under Kronecker products. We then continue to characterize probabilistic rank as the optimum of a covering linear program developed over the set of subtensors. This characterization establishes that probabilistic rank is a well-defined notion, and enables us to localize its analysis to the study of rank–weight ratios of individual subtensors in the important special case of tensors that admit a support-transitive group of permutation automorphisms, such as matrix multiplication tensors. Our main result in this section is a proof of Lemma 1.1, which enables our subsequent study of matrix multiplication tensors via templates.

3.1 Submultiplicativity. We state the lemma for probabilistic tensor rank, but a similar result holds for probabilistic border rank.

LEMMA 3.1. (SUBMULTIPLICATIVITY) *Let T_1 and T_2 be tensors of the same order. Then,*

$$\widetilde{\mathrm{rk}}\, T_1 \otimes T_2 \le \widetilde{\mathrm{rk}}\, T_1 \cdot \widetilde{\mathrm{rk}}\, T_2\,.$$

Proof. Let $\tilde{T}_1$ and $\tilde{T}_2$ be probabilistic tensors that support T_1 and T_2 elementwise with probabilities at least p_1 and p_2, respectively. Furthermore, assume that $\widetilde{\mathrm{rk}}\, T_1 = \mathbb{E}_{S_1\in\tilde{T}_1}[\mathrm{rk}\, S_1]p_1^{-1}$ and $\widetilde{\mathrm{rk}}\, T_2 = \mathbb{E}_{S_2\in\tilde{T}_2}[\mathrm{rk}\, S_2]p_2^{-1}$. Let $\tilde{T}$ be the probabilistic tensor defined by independently drawing two tensors $S_1 \in \tilde{T}_1$ and $S_2 \in \tilde{T}_2$ and then forming their Kronecker product $S = S_1 \otimes S_2$. Since S_1 and S_2 are independent, we observe that $\tilde{T}$ supports $T = T_1 \otimes T_2$ elementwise with probability at least $p = p_1 p_2$. Since tensor rank is submultiplicative and S_1, S_2 are independent, we observe that

$$\begin{aligned}
\widetilde{\mathrm{rk}}\, T_1 \otimes T_2 &\le \mathbb{E}_{S_1\otimes S_2\in\tilde{T}}[\mathrm{rk}\, S_1 \otimes S_2]p^{-1} \\
&\le \mathbb{E}_{S_1\otimes S_2\in\tilde{T}}[\mathrm{rk}\, S_1 \cdot \mathrm{rk}\, S_2]p^{-1} \\
&= \mathbb{E}_{S_1\in\tilde{T}_1}[\mathrm{rk}\, S_1]p_1^{-1}\mathbb{E}_{S_2\in\tilde{T}_2}[\mathrm{rk}\, S_2]p_2^{-1} \\
&= \widetilde{\mathrm{rk}}\, T_1 \cdot \widetilde{\mathrm{rk}}\, T_2\,.
\end{aligned}$$

3.2 Characterization by Linear Programming. Let us now characterize probabilistic rank through linear programming and tensor ranks of subtensors. This will in particular establish that the minimum in Definition 1.2 always exists. The characterization for probabilistic border rank is obtained in a similar way by replacing rank with border rank.

Let T be a nonzero tensor and let $S \in \operatorname{sub} T$ be a subtensor. For brevity, let us abbreviate $r_S = \mathrm{rk}\, S$. Let us associate a formal variable x_S with S, and similarly let us associate a formal variable y_{ijk} with every three-tuple $ijk \in \operatorname{supp} T$ in the support of T. Let us write S_{ijk} for the entry of S at position $ijk \in \operatorname{supp} T$.

Let us study the following linear programs. The primal program is to

$$\begin{aligned}
&\text{minimize } \textstyle\sum_{S\in\operatorname{sub} T} r_S x_S \\
&\text{subject to} \\
&\textstyle\sum_{S\in\operatorname{sub} T} [\![S_{ijk} \neq 0]\!] x_S \ge 1 \text{ for all } ijk \in \operatorname{supp} T\,, \\
&\qquad\qquad x_S \ge 0 \text{ for all } S \in \operatorname{sub} T\,.
\end{aligned} \tag{25}$$

The dual program is to

$$\begin{aligned} &\text{maximize } \textstyle\sum_{ijk\in\operatorname{supp} T} y_{ijk} \\ &\text{subject to} \\ &\textstyle\sum_{ijk\in\operatorname{supp} T} [\![S_{ijk}\neq 0]\!] y_{ijk} \le r_S \text{ for all } S\in \operatorname{sub} T\,, \\ &\qquad\qquad y_{ijk}\ge 0 \text{ for all } ijk\in\operatorname{supp} T. \end{aligned} \tag{26}$$

Let us first observe that any feasible solution x of the primal (25) defines a probability distribution over the subtensors in $\operatorname{sub} T$, and vice versa. Indeed, we can take the probability of $S\in\operatorname{sub} T$ to be

$$p_S = \frac{x_S}{\sum_{S\in\operatorname{sub} T} x_S} \tag{27}$$

and observe that $0\le p_S\le 1$ with $\sum_{S\in\operatorname{sub} T} p_S = 1$. Let $\tilde{T}$ be the probabilistic tensor for which the probability of S is p_S for all $S\in\operatorname{sub} T$. From (25) and (27) it follows that $\tilde{T}$ supports T elementwise with probability at least

$$p = \frac{1}{\sum_{S\in\operatorname{sub} T} x_S}\,.$$

Furthermore, we observe

$$\mathbb{E}_{S\in\tilde{T}}[\operatorname{rk} S]p^{-1} = \sum_{S\in\operatorname{sub} T} r_S p_S p^{-1} = \sum_{S\in\operatorname{sub} T} r_S x_S\,.$$

Thus, we have just shown:

LEMMA 3.2. (PROBABILISTIC RANK VIA LP) *For a nonzero tensor T, the optimum of the linear program (25) is exactly the probabilitic rank $\widetilde{\operatorname{rk}} T$.*

3.3 Support-Transitive Tensors. The objective of this section is to study the linear programs (25) and (26) when T is support-transitive and prove Lemma 1.1.

Let T be a nonzero tensor that admits a group G of permutation automorphisms that acts transitively on the support of T. In particular, the action of G extends to an action on $\operatorname{sub} T$ since for all $S\in\operatorname{sub} T$ it holds that $\operatorname{supp} S\subseteq\operatorname{supp} T$ and thus $\operatorname{supp} S^g = (\operatorname{supp} S)^g \subseteq (\operatorname{supp} T)^g = \operatorname{supp} T$ for all $g\in G$.

Select an arbitrary nonzero $S_0\in\operatorname{sub} T$. Since G consists of permutation automorphisms, we have $\operatorname{wt} S = \operatorname{wt} S_0$ and $\operatorname{rk} S = \operatorname{rk} S_0$ for all $S\in S_0^G$. For $ijk\in\operatorname{supp} T$, let us write P_{ijk} for the set of all tensors $S\in S_0^G$ with the property that $ijk\in\operatorname{supp} S$. Since G is support-transitive, we observe that for all $ijk, i'j'k'\in\operatorname{supp} T$ we have $|P_{ijk}| = |P_{i'j'k'}|$. Indeed, since there exists a $g\in G$ with $i'j'k' = ijk^g$, we have that $P_{i'j'k'} = P_{ijk}^g$. Let $L = |P_{ijk}|$.

Introduce the following feasible solution x to the primal program (25). For all $S\in\operatorname{sub} T$, set

$$x_S = \begin{cases} \frac{1}{L} & \text{if } S\in S_0^G\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{28}$$

In particular, we observe that the solution is feasible since for all $ijk\in\operatorname{supp} T$ it holds that $\sum_{S\in\operatorname{sub} T}[\![S_{ijk}\neq 0]\!]x_S = \sum_{S\in P_{ijk}} x_S = L\cdot\frac{1}{L} = 1$. Furthermore, the value of the solution x is

$$\frac{|S_0^G|\operatorname{rk} S_0}{L}\,. \tag{29}$$

Counting in two different ways the pairs (ijk, S) with $ijk\in\operatorname{supp} S$ and $S\in S_0^G$, we observe that

$$|S_0^G|\operatorname{wt} S_0 = L\operatorname{wt} T\,. \tag{30}$$

Thus, since the optimum of (25) is exactly the probabilistic rank of T, we conclude that

$$\widetilde{\operatorname{rk}} T \le \frac{|S_0^G|\operatorname{rk} S_0}{L} = \frac{\operatorname{wt} T\operatorname{rk} S_0}{\operatorname{wt} S_0}\,. \tag{31}$$

The inequality (7) in Lemma 1.1 now follows since the nonzero $S_0\in\operatorname{supp} T$ was arbitrary, and the case of the zero tensor is immediate.

To complete Lemma 1.1, it remains to establish that equality holds in (31) for at least one choice of $S_0\in\operatorname{supp} T$. Toward this end, let us study the dual program (26). For all $ijk\in\operatorname{supp} T$, set

$$y_{ijk} = \frac{\operatorname{rk} S_0}{\operatorname{wt} S_0}\,. \tag{32}$$

Thus, the dual program (26) is feasible with respect to (32) if and only if for all $S\in\operatorname{sub} T$ we have

$$\sum_{ijk\in\operatorname{supp} T} [\![S_{ijk}\neq 0]\!] y_{ijk} = \operatorname{wt} S\cdot\frac{\operatorname{rk} S_0}{\operatorname{wt} S_0} \le \operatorname{rk} S\,.$$

That is, the dual is feasible if and only if we have $\frac{\operatorname{rk} S_0}{\operatorname{wt} S_0}\le\frac{\operatorname{rk} S}{\operatorname{wt} S}$ for all $S\in\operatorname{sub} T$. That is, the dual is feasible if and only if the subtensor S_0 minimizes the rank-weight ratio among all subtensors of T. Assuming the dual solution (32) is feasible, we observe that its value is $\operatorname{wt} T\cdot\frac{\operatorname{rk} S_0}{\operatorname{wt} S_0}$. By (30) we observe that the dual value agrees with the primal value (29). Indeed,

$$|S_0^G|\cdot\frac{\operatorname{rk} S_0}{L} = \operatorname{wt} T\cdot\frac{\operatorname{rk} S_0}{\operatorname{wt} S_0}\,.$$

Thus, by linear programming duality we have that the primal solution (28) is the optimum of (25), and thus in particular we have

$$\widetilde{\operatorname{rk}} T = \operatorname{wt} T\cdot\frac{\operatorname{rk} S_0}{\operatorname{wt} S_0}$$

for exactly those subtensors $S_0\in\operatorname{sub} T$ that minimize the rank-weight ratio $\frac{\operatorname{rk} S_0}{\operatorname{wt} S_0}$. Again treating the zero tensor separately establishes that equality holds in (7)

for at least one subtensor. This completes the proof of Lemma 1.1. Furthermore, we now have a way of studying the probabilistic rank of a support-transitive tensor T by studying the ranks of its subtensors individually. Furthermore, this study may be restricted to representatives of orbits under the action of G. Conversely, we know that the probabilistic tensor that realizes the optimum probabilistic rank for a support-transitive tensor T can be without loss of generality assumed to be uniform distribution on an orbit of subtensors that realizes the optimum rank-weight ratio.

4 Templates and Circuits.

This section develops tools for working with subtensors of matrix multiplication tensors while simultaneously tracking the arithmetic complexity over the chosen field $\mathbb{F}$. The key observation is that the standard techniques for the complete tensor (the case $w = stu$) extend to proper subtensors (the case $w < stu$). We restrict the development in the present conference version of this paper to consider only tensor rank and associated templates, omitting border rank and its associated templates.

4.1 Arithmetic Circuits. A *circuit* is a directed acyclic graph whose nodes have either in-degree zero or in-degree two. The nodes of in-degree zero are called *input gates* and the nodes of in-degree two are called *arithmetic gates*. Each arithmetic gate is labeled either as an *addition* or as a *multiplication*. The two in-arcs to each addition gate further receive scalars from $\mathbb{F}$ as their labels.[17] Each input gate is labeled with a distinct variable, such as an entry of a matrix, that enables one to feed input to a circuit or to compose circuits. One or more gates of a circuit are designated as *output gates*, again labeled with variables to enable reading output or circuit composition.

A circuit is a *sum–product–sum* circuit if every directed path from an input gate to an output gate contains exactly one multiplication gate.

4.2 An Example Circuit Template. Recall the definition of a circuit template with parameters $\langle s,t,u|m,w\rangle^{\bar{a},\bar{b},\bar{c}}_{\underline{a},\underline{b},\underline{c}}$ (Definition 1.4).

Let us illustrate circuit templates with a concrete $\langle 2|7\rangle^4_7$-circuit template obtained from the Strassen–Winograd algorithm. Below in (33), the leftmost column displays the matrices of the three groups of linear forms α,β,γ that define a $\langle 2|7\rangle$-template, which can be readily verified by substitution to (15) and witnessing that $T = \langle 2,2,2\rangle$ results. To obtain a $\langle 2|7\rangle^4_7$-circuit template, we need to realize each of these three groups of linear forms using a circuit with 4 addition gates; these realizations are given in the middle column. Furthermore, we need to realize each of the three transposed groups of linear forms using a circuit with 7 addition gates; these realizations are given in the rightmost column.

(33)

	x_{11}	x_{12}	x_{21}	x_{22}
α_1	0	0	1	1
α_2	0	1	0	0
α_3	0	1	0	1
α_4	0	1	1	1
α_5	1	1	1	1
α_6	0	0	1	0
α_7	1	0	0	0

$\alpha_1 = x_{21} + x_{22}$
$\alpha_2 = x_{12}$
$\alpha_3 = x_{12} + x_{22}$
$\alpha_4 = \alpha_3 + x_{21}$
$\alpha_5 = \alpha_4 + x_{11}$
$\alpha_6 = x_{21}$
$\alpha_7 = x_{11}$

$x_{11} = \alpha_5 + \alpha_7$
$t_1 = \alpha_4 + \alpha_5$
$t_2 = \alpha_3 + t_1$
$x_{12} = \alpha_2 + t_2$
$t_3 = \alpha_1 + t_1$
$x_{21} = \alpha_6 + t_3$
$x_{22} = \alpha_1 + t_2$

	y_{11}	y_{12}	y_{21}	y_{22}
β_1	0	0	1	1
β_2	0	0	1	0
β_3	0	1	0	−1
β_4	0	−1	1	1
β_5	0	1	0	0
β_6	1	1	−1	−1
β_7	1	0	0	0

$\beta_1 = y_{21} + y_{22}$
$\beta_2 = y_{21}$
$\beta_3 = y_{12} - y_{22}$
$\beta_4 = y_{21} - \beta_3$
$\beta_5 = y_{12}$
$\beta_6 = y_{11} - \beta_4$
$\beta_7 = y_{11}$

$y_{11} = \beta_6 + \beta_7$
$t_1 = \beta_4 - \beta_6$
$t_2 = \beta_1 + t_1$
$t_3 = \beta_3 + \beta_5$
$y_{12} = t_3 - t_1$
$y_{21} = t_2 + \beta_2$
$y_{22} = t_2 - \beta_3$

	z_{11}	z_{12}	z_{21}	z_{22}
γ_1	0	0	−1	1
γ_2	1	−1	−1	1
γ_3	0	1	0	−1
γ_4	0	1	1	−1
γ_5	0	0	1	0
γ_6	0	1	0	0
γ_7	1	0	0	0

$\gamma_1 = z_{22} - z_{21}$
$\gamma_4 = z_{12} - \gamma_1$
$\gamma_2 = z_{11} - \gamma_4$
$\gamma_3 = z_{12} - z_{22}$
$\gamma_5 = z_{21}$
$\gamma_6 = z_{12}$
$\gamma_7 = z_{11}$

$z_{11} = \gamma_2 + \gamma_7$
$t_1 = \gamma_2 - \gamma_4$
$t_2 = \gamma_1 + t_1$
$t_3 = \gamma_3 - t_1$
$z_{12} = \gamma_6 + t_3$
$z_{21} = \gamma_5 - t_2$
$z_{22} = t_2 - \gamma_3$

We can also immediately obtain a $\langle 2|6,7\rangle^4_6$-circuit template from (33) by deleting the forms α_7, β_7, and γ_7. In this case the three groups α,β,γ realize (19).

4.3 Coarse-Grained Realization. This section develops coarse-grained parameters for realizing a $\langle s,t,u|m,w\rangle$-template as a circuit template. Accordingly, let $S \in \mathbb{F}[x,y,z]$ be a subtensor of $\langle s,t,u\rangle$ of weight w and let the linear forms $\alpha_\ell(x) \in \mathbb{F}[x], \beta_\ell(y) \in \mathbb{F}[y], \gamma_\ell(z) \in \mathbb{F}[z]$ for $\ell = 1,2,\ldots,m$ represent S in the sense of (15).

Using the procedure illustrated in §2.3 and in (33), with the forms α and β guiding how to take sums of the entries of the $s \times t$ and $t \times u$ left and right inputs, respectively, to obtain the inputs to each of the m multiplication gates, and the transposed forms γ guiding how to sum the m products to obtain the entries of the

[17] With the semantics that an addition gate that takes input from gates with values x and y via arcs with scalar labels α and β, respectively, evaluates to value $\alpha x + \beta y$. This in particular enables individual addition gates to capture negation and subtraction as appropriate.

$s \times u$ product, we observe that a template can always be turned into a sum–product–sum circuit that evaluates the bilinear map $\mathbb{F}^{s\times t} \times \mathbb{F}^{t\times u} \to \mathbb{F}^{s\times u}$ defined by S using at most m multiplication gates and the following budget of addition gates: (i) at most $m(st-1)$ addition gates to prepare the $s \times t$ matrix for multiplication, (ii) at most $m(tu-1)$ addition gates to prepare the $t \times u$ matrix for multiplication, and (iii) at most $su(m-1)$ addition gates to postprocess the results of multiplication to the $s \times u$ output. By symmetry, we have just shown that:

LEMMA 4.1. (COARSE-GRAINED REALIZATION) *Suppose that there exists a template with parameters $\langle s,t,u|m,w\rangle$. Then, there exists a circuit template with parameters $\langle s,t,u|m,w\rangle^{m(st-1),\,m(tu-1),\,m(us-1)}_{st(m-1),\,tu(m-1),\,us(m-1)}$.*

Or, what is even more coarse-grained, an $\langle s,t,u|m,w\rangle$-template implies a circuit template with parameters $\langle s,t,u|m,w\rangle^{mstu}_{mstu}$. These parameters will suffice for asymptotic results such as Theorem 1.1.

4.4 Tools for Circuit Templates. Let us now prepare a set of lemmas for working with circuit templates. Our key tool will be template composition. The following lemma is immediate by symmetry in the definitions.

LEMMA 4.2. (ROTATION) *Suppose that there exits a circuit template with parameters $\langle s,t,u|m,w\rangle^{\bar a,\bar b,\bar c}_{\underline a,\underline b,\underline c}$. Then, there exists a circuit template with parameters $\langle t,u,s|m,w\rangle^{\bar b,\bar c,\bar a}_{\underline b,\underline c,\underline a}$.*

The next lemma gives fine-grained control over template composition. We omit the proof from the present conference version of this paper.

LEMMA 4.3. (COMPOSITION) *Suppose there exist circuit templates with parameters $\langle s_1,t_1,u_1|m_1,w_1\rangle^{\bar a_1,\bar b_1,\bar c_1}_{\underline a_1,\underline b_1,\underline c_1}$ and $\langle s_2,t_2,u_2|m_2,w_2\rangle^{\bar a_2,\bar b_2,\bar c_2}_{\underline a_2,\underline b_2,\underline c_2}$. Then, there exists a circuit template with parameters $\langle s,t,u|m,w\rangle^{\bar a,\bar b,\bar c}_{\underline a,\underline b,\underline c}$ where*

$$\begin{aligned} s &= s_1 s_2 & \bar a &= \bar a_1 s_2 t_2 + m_1 \bar a_2 \\ t &= t_1 t_2 & \bar b &= \bar b_1 t_2 u_2 + m_1 \bar b_2 \\ u &= u_1 u_2 & \bar c &= \bar c_1 u_2 s_2 + m_1 \bar c_2 \\ m &= m_1 m_2 & \underline a &= \underline a_1 s_2 t_2 + m_1 \underline a_2 \\ w &= w_1 w_2 & \underline b &= \underline b_1 t_2 u_2 + m_1 \underline b_2 \\ & & \underline c &= \underline c_1 u_2 s_2 + m_1 \underline c_2 \end{aligned} \tag{34}$$

Using rotation and composition, we obtain:

LEMMA 4.4. (SYMMETRIZATION) *Suppose that there exists a template with parameters $\langle s,t,u|m,w\rangle$. Then, there exists a template with parameters $\langle stu,stu,stu|m^3,w^3\rangle$.*

We conclude this section by recording the following detailed description of one of the possible maps that can be instantiated as a circuit from a circuit a template. In particular it is immediate that we can localize a description of the map to the set W. In what follows we will introduce randomization to effectively randomize this set.

LEMMA 4.5. (MAP GIVEN BY A CIRCUIT TEMPLATE) *An $\langle s,t,u|m,w\rangle^{\bar a,\bar b,\bar c}_{\underline a,\underline b,\underline c}$-circuit template in particular defines a circuit that computes from given inputs $F \in \mathbb{F}^{s\times t}$ and $G \in \mathbb{F}^{t\times u}$ the output $H \in \mathbb{F}^{s\times u}$ using at most $\bar a + \bar b + \underline c$ additions and at most m multiplications of scalars in $\mathbb{F}$. Moreover, there exists a set $W \subseteq [s] \times [t] \times [u]$ of size exactly w such that for all $i \in [s]$ and $k \in [u]$ it holds that*

$$H_{ik} = \sum_{\substack{j\in[t] \\ (i,j,k)\in W}} F_{ij} G_{jk}\,. \tag{35}$$

Lemma 4.5 thus in particular gives a matrix multiplication algorithm over $\mathbb{F}$ when $w = stu$, and a broken matrix multiplication algorithm when $w < stu$.

5 Probabilistic Rank of Matrix Multiplication.

This section studies the probabilistic rank of matrix multiplication tensors with a focus on lower bounds. We start by deriving an elementary lower bound on rank of subtensors, which we can then transport using Theorem 1.4 and template tools to a lower bound on probabilistic rank. Our main result in this section is a partial asymptotic derandomization of probabilistic rank over the complex field, given in Theorem 1.3 in the introduction. Over finite fields, we do not have any better lower bounds for probabilistic rank other than what is given by the general lower bound in Theorem 5.1.

5.1 Rank of Subtensors. The following elementary lemma holds for both rank and border rank.

LEMMA 5.1. *Any subtensor S of $\langle s,s,s\rangle$ satisfies $\operatorname{rk} S \geq \operatorname{wt} S/s$.*

Proof. Recall from §2.4 that

$$\operatorname{supp}\langle s,s,s\rangle = \{(ij,jk,ki) : i \in [s], j \in [s], k \in [s]\}\,.$$

Partition $\operatorname{supp}\langle s,s,s\rangle$ to s parts $P_1, P_2, \ldots, P_s$ with

$$P_\ell = \{(ij,jk,ki) : i \in [s], j \in [s], k \in [s], i+j \equiv \ell \bmod s\}.$$

Since S has weight $\operatorname{wt} S$, there exists a part P_ℓ such that $|\operatorname{supp} S \cap P_\ell| \geq \operatorname{wt} S/s$. Sum the $s^2 \times s^2 \times s^2$ tensor S along the first mode (indexed by ij) to an $s^2 \times s^2$ matrix and observe that this matrix is a partial permutation matrix with $\operatorname{wt} S/s$ ones, and thus has rank $\operatorname{wt} S/s$.

5.2 A Lower Bound for Probabilistic Rank. The following lower bound holds for both probabilistic rank and probabilistic border rank.

THEOREM 5.1. *For all integers s, t, u, we have*

$$\widetilde{\mathrm{rk}}\,\langle s,t,u\rangle \geq (stu)^{2/3}\,.$$

Proof. From Theorem 1.4 it follows that there exists an $\langle s,t,u|m,w\rangle$-template with $\widetilde{\mathrm{rk}}\,\langle s,t,u\rangle = mstu/w$. By symmetrization, we conclude that there exists a template with parameters $\langle stu,stu,stu|m^3,w^3\rangle$. In particular, the subtensor S of $\langle stu,stu,stu\rangle$ with weight w^3 and rank at most m^3 guaranteed by the template must by Lemma 5.1 satisfy $\frac{w^3}{stu} \leq \mathrm{rk}\,S \leq m^3$. Rearranging and taking cube roots, we conclude that $\frac{m}{w} \geq (stu)^{-1/3}$.

5.3 A Lower Bound via Support Rank. This section proves Theorem 1.3. That is, over the complex field, we must show that $\omega_{\mathrm{s}} \leq \inf\{\tau : \widetilde{\mathrm{rk}}\,\langle t,t,t\rangle = O(t^\tau)\}$. We proceed by a variant of Adleman's argument [3] to construct a collection tensors that together cover the support of $\langle t,t,t\rangle$. From Theorem 1.4 it follows that there exists an $\langle t,t,t|m,w\rangle$-template with $\widetilde{\mathrm{rk}}\,\langle t,t,t\rangle = mt^3/w$. Let us transform this template to the probabilistic tensor $\tilde{T}$ that realizes the probabilistic rank (cf. §3.3). This probabilistic tensor has the property that $\tilde{T}$ supports $\langle t,t,t\rangle$ entrywise with probability $p = w/t^3$. Let us now draw r independent tensors $S_1, S_2, \ldots, S_r \in \tilde{T}$ from the distribution $\tilde{T}$. All these tensors satisfy $\mathrm{rk}\,S_\ell = m$ and $\mathrm{wt}\,S_\ell = w$. Each outcome S_ℓ is a subtensor of $\langle t,t,t\rangle$. Select an arbitrary entry $ijk \in \mathrm{supp}\,\langle t,t,t\rangle$ and study the probability ϵ that none of the r outcomes satisfies $ijk \in \mathrm{supp}\,S_\ell$. By independence, we have $\epsilon \leq (1-p)^r \leq \exp(-pr)$. We want $t^3\epsilon < 1$ so that there is at least outcome for the r tensors $S_1, S_2, \ldots, S_r$ where for every $ijk \in \mathrm{supp}\,\langle t,t,t\rangle$ there exists a tensor S_ℓ with $ijk \in \mathrm{supp}\,S_\ell$. It suffices to take $r = p^{-1}(\delta + 3\ln t)$ where $\delta > 0$ is the least positive constant such that r is an integer. Over the complex field it is possible to find coefficients $\kappa_1, \kappa_2, \ldots, \kappa_r$ such that the tensor $S = \sum_{\ell=1}^r \kappa_\ell S_\ell$ satisfies $\mathrm{supp}\,S = \mathrm{supp}\,\langle t,t,t\rangle$. Indeed, observe that finding these coefficients amounts to finding in an r-dimensional complex vector space a point *not* on any hyperplane in a set of t^3 hyperplanes through the origin. Next observe that we have

$$\mathrm{rk}\,S \leq mr = \frac{mt^3}{w}(\delta + 3\ln t) = (\delta + 3\ln t)\,\widetilde{\mathrm{rk}}\,\langle t,t,t\rangle$$

and thus $\mathrm{rk}_{\mathrm{s}}\,\langle t,t,t\rangle \leq \mathrm{rk}\,S \leq (\delta + 3\ln t)\,\widetilde{\mathrm{rk}}\,\langle t,t,t\rangle$. Taking logarithms on both sides, we have

$$\log_t \mathrm{rk}_{\mathrm{s}}\,\langle t,t,t\rangle \leq \frac{\log(\delta + 3\ln t)}{\log t} + \log_t \widetilde{\mathrm{rk}}\,\langle t,t,t\rangle\,.$$

The inequality (9) now follows by letting t grow without bound. We observe that this proof fails over a finite field because finding the coefficients $\kappa_1, \kappa_2, \ldots, \kappa_r$ is no longer straightforward. Thus, it appears that derandomizing probabilistic rank over a finite field is less straightforward. In applications such as Boolean matrix multiplication in §6.6, it is possible to rely on an analogous argument over an arbitrary field by introducing a formal indeterminate δ that keeps the outcomes distinct from each other.

6 Boolean Matrix Multiplication.

We now prove our main theorems for Boolean matrix multiplication, namely Theorem 1.1 and Theorem 1.5.

6.1 Randomizing a Template. We start with the proof of Theorem 1.5. Suppose that there exists a circuit template with parameters $\langle s,t,u|m,w\rangle^{\bar{a},\bar{b},\bar{c}}_{\underline{a},\underline{b},\underline{c}}$. First, let us introduce randomization to the setting of Lemma 4.5. Suppose we are given $A \in \mathbb{F}^{s\times t}$ and $B \in \mathbb{F}^{t\times u}$ as input. Draw independently and uniformly at random three permutations, $\sigma \in \mathrm{Sym}([s])$, $\tau \in \mathrm{Sym}([t])$, and $\upsilon \in \mathrm{Sym}([u])$. Let $F = A^{\sigma,\tau}$ be the matrix obtained from A by permuting the rows according to σ and the columns according to τ. In precise terms, let $F_{ij} = A_{i^\sigma j^\tau}$ for all $i \in [s]$ and $j \in [t]$. Introduce the matrix $G = B^{\tau,\upsilon}$ in a similar way. Use Lemma 4.5 to compute the output matrix H. In particular, recall that there is a fixed set $W \subseteq [s]\times[t]\times[u]$ of size W such that for all $i \in [s]$ and $k \in [u]$ it holds that $H_{ik} = \sum_{j\in[t]:(i,j,k)\in W} F_{ij}G_{jk}$. Permute the output using the inverse permutations to obtain the matrix $C = H^{\sigma^{-1},\upsilon^{-1}}$.

The matrices A, B, and C now satisfy, for all $i \in [s]$ and $k \in [u]$,

$$C_{ik} = \sum_{\substack{j\in[t]\\(i,j,k)\in W^{\sigma,\tau,\upsilon}}} A_{ij}B_{jk}\,, \tag{36}$$

where $W^{\sigma,\tau,\upsilon} = \{(i^\sigma, j^\tau, k^\upsilon) : (i,j,k) \in W\}$ is a random set with the following key property. For all $(i,j,k) \in [s]\times[t]\times[u]$ it holds that

$$\Pr_{\sigma,\tau,\upsilon}[(i,j,k) \in W^{\sigma,\tau,\upsilon}] = \frac{w}{stu}\,. \tag{37}$$

We remark in passing that an alternative, more cache-friendly randomization strategy is possible when the template is a composition of d templates. With this strategy, at each level $\ell = 1,2,\ldots,d$ of the composition one applies independent random permutations $\sigma_\ell, \tau_\ell, \upsilon_\ell$ to each component template in the composition to obtain the property (37); cf. (5) for an example. In this case it should be emphasized that the aggregate

permutations σ, τ, υ are not uniform random permutations, but (37) holds for the set W. Let us now turn this randomized map to an algorithm for sketching Boolean products, and the proof of Theorem 1.5.

6.2 Sketching over the Binary Field. Let $U \in \{0,1\}^{s\times t}$ and $V \in \{0,1\}^{t\times u}$ be Boolean matrices. We seek to compute the Boolean product $P \in \{0,1\}^{s\times u}$ defined for all $i \in [s]$ and $k \in [u]$ by $P_{ik} = \bigvee_{j\in[t]} U_{ij} \wedge V_{jk}$. Let us use (36) over the binary field $\mathbb{F}_2$. Set $A = U$. Construct the matrix B from the matrix V by first setting $B = V$ and then assigning each 1-entry of B to 0 independently with probability $1/2$. Now compute the matrix C from the matrices A and B in (36), working over $\mathbb{F}_2$.

Let us first observe that for all $i \in [s]$ and $k \in [u]$ we have that $P_{ik} = 0$ implies $C_{ik} = 0$ with probability 1. This establishes (i) of Theorem 1.5.

Next, we claim that $P_{ik} = 1$ implies $C_{ik} = 1$ with probability at least $\frac{w}{2stu}$. To see this, let $J = \{j \in [t] : U_{ij} \wedge V_{jk} = 1\}$ and observe that J is nonempty since $P_{ik} = 1$. Select an arbitrary $j_0 \in J$. Let us condition on the event $(i, j_0, k) \in W^{\sigma,\tau,\upsilon}$, which by (37) has probability at least $\frac{w}{stu}$. Given $(i, j_0, k) \in W^{\sigma,\tau,\upsilon}$, from our randomization of B it follows that (36) is a nonempty sum of independent uniformly distributed random variables in $\mathbb{F}_2$, and thus $C_{ij} = 1$ with probability $1/2$. The claim follows. This completes the proof of Theorem 1.5.

6.3 Sketching over Other Fields. Let us observe in passing that a similar approach clearly works over any field of characteristic 2. Indeed, for fields of characteristic other than 2, we can change the implication in (ii) of Theorem 1.5 from $R_{ik} = 1$ to $R_{ik} \neq 0$ and detect this without decreasing the witnessing probability from the claimed $\frac{w}{2stu}$.

6.4 Coarse-Grained Analysis. This section starts work towards our coarse-grained theorem for Boolean matrix multiplication (Theorem 1.1) establishing that the probabilistic rank of $\langle s,t,u\rangle$ controls the complexity of Boolean matrix multiplication from above. We split the analysis into two cases, namely that of *noncontractive* and *contractive* templates. Fix integers $s,t,u \geq 2$ and the constant $\epsilon > 0$. From Theorem 1.4 we obtain that there exists an $\langle s,t,u|m,w\rangle$-template with $\widetilde{\mathrm{rk}}\,\langle s,t,u\rangle = mstu/w$.

6.5 Noncontractive Template. Let us first assume that the template satisfies the inequality

$$m \geq (stu)^{2/3}\,.$$

In this case we say that the template is *noncontractive*.

Let us symmetrize the $\langle s,t,u|m,w\rangle$-template to a $\langle q,q,q|m^3,w^3\rangle$-template with $q = stu$. Observe in particular that q is a constant. Take a coarse-grained realization of the template to obtain a circuit template with parameters $\langle q,q,q|m^3,w^3\rangle^C_C$ for a constant C with $C \leq m^3q^3 = m^3s^3t^3u^3$.

Suppose we are given as input two Boolean matrices of shape $n \times n$ with $n \geq 2$. Let $d = \lceil \log_q n\rceil$ and use Lemma 1.2 on the template with parameters $\langle q,q,q|m^3,w^3\rangle^C_C$ to obtain a template with parameters $\langle q^d,q^d,q^d|m^{3d},w^{3d}\rangle^{CD}_{CD}$, where

$$D = \begin{cases} dq^{2d} & \text{if } m^3 = q^2,\\ \frac{m^{3d}-q^{2d}}{m^3-q^2} & \text{if } m^3 \neq q^2. \end{cases} \tag{38}$$

Let us write $\delta_{stu} = [\![m = (stu)^{2/3}]\!] = [\![m^3 = q^2]\!]$. Since the template is noncontractive with $m^3 \geq q^2$, we have

$$D = O(m^{3d}\log^{\delta_{stu}} n)\,. \tag{39}$$

Pad the input matrices with zeros so that they have shape $q^d \times q^d$ and run the sketching algorithm from Theorem 1.5 (over the field $\mathbb{F}$) for $r = \lceil 3(q^{3d}/w^{3d})\ln n\rceil$ repetitions to recover the Boolean product with high probability. Since each run of the sketching algorithm costs $O(m^{3d}\log^{\delta_{stu}} n)$ operations in $\mathbb{F}$, the total number of operations in $\mathbb{F}$ is $O\big(\big(\frac{mq}{w}\big)^{3d}\log^{1+\delta_{stu}} n\big) = O\big(n^{\log_{(stu)^{1/3}}\widetilde{\mathrm{rk}}\,\langle s,t,u\rangle}\log^{1+\delta_{stu}} n\big)$. This proves Theorem 1.1 subject to the assumption that the template is noncontractive.

6.6 Contractive Template. Let us now assume that the $\langle s,t,u|m,w\rangle$-template is *contractive* with $m < (stu)^{2/3}$; that is, with $m^3 < q^2$ after symmetrization to $\langle q,q,q|m^3,w^3\rangle$ with $q = stu$. From Lemma 5.1 applied to $\langle q,q,q|m^3,w^3\rangle$ we observe that

$$\frac{w^3}{q} \leq m^3\,. \tag{40}$$

In particular, our assumption $m^3 < q^2$ and $w = q$ and lead to a contradiction, which implies $w < q$.

The main technical obstacle when working with a contractive template is that (39) no longer holds, but rather we have $D = \Omega(d^{2d})$, so it is asymptotically too expensive to use the template as in the noncontractive case. To work around this obstacle, we proceed with the following strategy. First, since $w < q$, we can amplify the gap between the parameters w and q using Lemma 1.2 to boost the template to its b^{th} power $\langle q^b,q^b,q^b|m^{3b},w^{3b}\rangle$ for a positive integer constant b whose value we will fix later. We will then use a collection of k permuted copies of the boosted template so

that (i) the collection as a whole becomes noncontractive with $km^{3b} > q^{2b}$, and (ii) the union of the supports of the collection has size W with

$$W \geq q^{3b}\left(1-\left(1-\left(\frac{w}{q}\right)^{3b}\right)^{k}\right). \tag{41}$$

Here k is again a positive integer constant whose value we will fix later.

Let us now establish that such a collection of k permuted copies of the boosted template exists, where by *permutation* we mean acting on the boosted template with a three-tuple of independent permutations $\sigma, \tau, \upsilon \in \mathrm{Sym}([q^b])$ as in (24). Suppose we select k such three-tuples of permutations independently and uniformly at random, and act on the template $\langle q^b, q^b, q^b | m^{3b}, w^{3b}\rangle$ with each three-tuple to obtain a collection of k permuted templates. The probability that any fixed entry in the support of $\langle q^b, q^b, q^b\rangle$ occurs in the support of at least one of the k permuted templates is $1-\left(1-\frac{w^{3b}}{q^{3b}}\right)^k$. Thus, the expected number of entries in the support of $\langle q^b, q^b, q^b\rangle$ that occur in the support of at least one of the k permuted templates is equal to the right-hand side of (41). In particular, there exists at least one collection of k permuted templates for which (ii) holds. Set

$$k = \left\lfloor \frac{q^{2b}}{m^{3b}} \right\rfloor + 1 \tag{42}$$

so that (i) holds, and let $0 < \kappa \leq 1$ so that $k = \left(\frac{q^2}{m^3}\right)^b + \kappa$. Since $m^3 < q^2$, we can make k arbitrarily large by increasing b.

Without yet fixing the value of b, suppose the template $\langle q^b, q^b, q^b | m^{3b}, w^{3b}\rangle$ realizes via (15) a subtensor S of $\langle q^b, q^b, q^b\rangle$ of weight w^{3b} using the forms $\alpha_1, \alpha_2, \ldots, \alpha_{m^{3b}}$, $\beta_1, \beta_2, \ldots, \beta_{m^{3b}}$, and $\gamma_1, \gamma_2, \ldots, \gamma_{m^{3b}}$.

For $j = 1, 2, \ldots, k$, let us write S_j for the j^{th} permuted version of S in a collection that satisfies (i) and (ii). Similarly, let us write $\alpha_1^{(j)}, \alpha_2^{(j)}, \ldots, \alpha_{m^{3b}}^{(j)}$, $\beta_1^{(j)}, \beta_2^{(j)}, \ldots, \beta_{m^{3b}}^{(j)}$, and $\gamma_1^{(j)}, \gamma_2^{(j)}, \ldots, \gamma_{m^{3b}}^{(j)}$ for the corresponding forms that realize S_j.

To control the collection $S_1, S_2, \ldots, S_k$ over an arbitrary field $\mathbb{F}$, we will extend from $\mathbb{F}$ to $\mathbb{F}_\delta = \mathbb{F}[\delta]$ for an indeterminate δ, and eventually use truncated polynomial arithmetic to return to $\mathbb{F}$. Let

$$S_\delta = S_1\delta + S_2\delta^2 + \ldots + S_k\delta^k$$

and, for all $\ell = 1, 2, \ldots, m^{3b}$,

$$\hat{\gamma}_\ell^{(1)} = \gamma_\ell^{(1)}\delta, \quad \hat{\gamma}_\ell^{(2)} = \gamma_\ell^{(2)}\delta^2, \quad \ldots, \quad \hat{\gamma}_\ell^{(k)} = \gamma_\ell^{(k)}\delta^k.$$

We can thus realize S_δ as $S_\delta = \sum_{j=1}^{k}\sum_{\ell=1}^{m^{3b}} \alpha_\ell^{(j)}\beta_\ell^{(j)}\hat{\gamma}_\ell^{(j)}$. Let us write $\langle q^b, q^b, q^b | km^{3b}, W \| k\rangle_C^C$ for a circuit template over $\mathbb{F}_\delta$ that gives a coarse-grained realization of S_δ using intermediate results of δ-degree at most k via the forms $\alpha_1^{(j)}, \alpha_2^{(j)}, \ldots, \alpha_{m^{3b}}^{(j)}$, $\beta_1^{(j)}, \beta_2^{(j)}, \ldots, \beta_{m^{3b}}^{(j)}$, and $\hat{\gamma}_1^{(j)}, \hat{\gamma}_2^{(j)}, \ldots, \hat{\gamma}_{m^{3b}}^{(j)}$ for $j = 1, 2, \ldots, k$. In particular, C is a constant that depends only on the constants s, t, u, m, w, k, b.

Suppose we are given as input two Boolean matrices of shape $n \times n$ with $n \geq 2$. Let $d = \lceil \log_{q^b} n \rceil$ and use Lemma 1.2 on the template with parameters $\langle q^b, q^b, q^b | km^{3b}, W \| k\rangle_C^C$ to obtain a template with parameters $\langle q^{bd}, q^{bd}, q^{bd} | (km^{3b})^d, W^d \| dk\rangle_{CD}^{CD}$, where

$$D = \begin{cases} dq^{2bd} & \text{if } km^{3b} = q^{2b}, \\ \frac{(km^{3b})^d - q^{2bd}}{km^{3b} - q^{2b}} & \text{if } km^{3b} \neq q^{2b}. \end{cases}$$

Since the template is noncontractive with $km^{3b} > q^{2b}$, we thus have $D = O\big((km^{3b})^d\big)$.

Pad the input matrices with zeros so that they have shape $q^{bd} \times q^{bd}$ and run the randomized sketching algorithm from §6.2 and §6.3 using the template $\langle q^{bd}, q^{bd}, q^{bd} | (km^{3b})^d, W^d \| dk\rangle_{CD}^{CD}$ over $\mathbb{F}_\delta$ for $r = \lceil 3(q^{3bd}/W^d)\ln n \rceil$ repetitions. Each sketch obtained is a $q^{bd} \times q^{bd}$ matrix over $\mathbb{F}_\delta$, where each entry is a polynomial in δ of degree at most dk. Taking the union of the supports of the sketches, we recover the Boolean product with high probability. Indeed, the template $\langle q^{bd}, q^{bd}, q^{bd} | (km^{3b})^d, W^d \| dk\rangle_{CD}^{CD}$ realizes $S_\delta^{\otimes d}$ with support size W^d.

Using truncated polynomial arithmetic to implement the operations in $\mathbb{F}_\delta$, each run of the sketching algorithm costs $O((km^{3b})^d(dk)^2)$ operations in $\mathbb{F}$. The total number of operations in $\mathbb{F}$ is thus

$$O\left(\frac{(km^{3b}q^{3b})^d}{W^d}\log^2 n\right). \tag{43}$$

Let us now analyse the growth rate of (43) as a polynomial in n. From (41) and (42), we have

$$\frac{km^{3b}q^{3b}}{W} \leq \frac{km^{3b}q^{3b}}{q^{3b}(1-(1-(\frac{w}{q})^{3b})^k)} \leq \frac{q^{2b}+\kappa m^{3b}}{1-(1-(\frac{w}{q})^{3b})^k}. \tag{44}$$

Recalling that $1 + x \leq e^x = \sum_{j=0}^{\infty}\frac{x^j}{j!}$ for all real x, we have

$$1-\left(1-\left(\frac{w}{q}\right)^{3b}\right)^k \geq 1-e^{-k(\frac{w}{q})^{3b}}. \tag{45}$$

Let us next observe from (42) that

$$k\left(\frac{w}{q}\right)^{3b} = \left(\frac{w^3}{m^3q}\right)^b + \kappa\left(\frac{w^3}{q^3}\right)^b. \tag{46}$$

Since $m^3 < q^2$ by contractivity, we have $\frac{w^3}{m^3q} > \frac{w^3}{q^3}$. Recalling from (40) that $1 \geq \frac{w^3}{m^3q}$, we split into two cases based on whether equality holds in the inequality.

First, suppose that $1 = \frac{w^3}{m^3 q}$. In this case we have $\widetilde{\mathrm{rk}}\,\langle s,t,u\rangle = \frac{mq}{w} = q^{2/3}$. Then, from (46) we have that $k\big(\frac{w}{q}\big)^{3b} \geq 1$. For all large enough b we thus have from (44) and (45) that $km^{3b}q^{3b}/W \leq 2q^{2b}/(1-e^{-1})$. Thus, (43) is bounded by $O\big(n^{2+\frac{1}{b}\log_q \frac{2}{1-e^{-1}}}\log^2 n\big)$. That is, for a large enough constant b, we have that (43) is bounded by $O\big(n^{2+\epsilon}\big) = O\big(n^{(\log_{(stu)^{1/3}} \widetilde{\mathrm{rk}}\,\langle s,t,u\rangle)+\epsilon}\big)$.

Second, suppose that $1 > \frac{w^3}{m^3 q}$. From (45), we have

$$\begin{aligned} 1-\big(1-\big(\frac{w}{q}\big)^{3b}\big)^k &\geq \sum_{j=1}^{\infty}(-1)^{j+1}\frac{(k(\frac{w}{q})^{3b})^j}{j!} \\ &\geq k\big(\frac{w}{q}\big)^{3b} - \frac{(k(\frac{w}{q})^{3b})^2}{1-k(\frac{w}{q})^{3b}}\,. \end{aligned} \tag{47}$$

Thus, since $1 > \frac{w^3}{m^3 q} > \frac{w^3}{q^3}$, from (46), (47), and (44) it follows that for all large enough b we have

$$\frac{km^{3b}q^{3b}}{W} \leq \frac{2q^{2b}}{\frac{1}{2}\big(\frac{w^3}{m^3 q}\big)^b} \leq 4\bigg(\frac{m^3q^3}{w^3}\bigg)^b\,.$$

Thus, (43) is bounded by $O\big(n^{\log_q \frac{m^3q^3}{w^3}+\frac{1}{b}\log_q 4}\log^2 n\big)$. For a large enough constant b, (43) is thus bounded by $O\big(n^{(\log_{q^{1/3}} \frac{mq}{w})+\epsilon}\big) = O\big(n^{(\log_{(stu)^{1/3}} \widetilde{\mathrm{rk}}\,\langle s,t,u\rangle)+\epsilon}\big)$. This proves Theorem 1.1 subject to the assumption that the template is contractive.

Acknowledgement.

We are grateful to Andreas Björklund for useful discussions and to the anonymous reviewers for their remarks that helped to improve this paper.

References

[1] *19th Annual Symposium on Foundations of Computer Science, Ann Arbor, Michigan, USA, 16-18 October 1978*. IEEE Computer Society, 1978.

[2] A. Abboud, R. R. Williams, and H. Yu. More applications of the polynomial method to algorithm design. In Indyk [44], pages 218–230.

[3] L. M. Adleman. Two theorems on random polynomial time. In *19th Annual Symposium on Foundations of Computer Science, Ann Arbor, Michigan, USA, 16-18 October 1978* [1], pages 75–83.

[4] J. Alman and R. Williams. Probabilistic polynomials and Hamming nearest neighbors. In V. Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 136–150. IEEE Computer Society, 2015.

[5] J. Alman and R. R. Williams. Probabilistic rank and matrix rigidity. In H. Hatami, P. McKenzie, and V. King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 641–652. ACM, 2017.

[6] V. L. Arlazarov, E. A. Dinic, M. A. Kronrod, and I. A. Faradzhev. On economical construction of the transitive closure of an oriented graph. *Sov. Math., Dokl.*, 11:1209–1210, 1970.

[7] G. Ballard, J. Demmel, O. Holtz, B. Lipshitz, and O. Schwartz. Communication-optimal parallel algorithm for Strassen's matrix multiplication. In G. E. Blelloch and M. Herlihy, editors, *24th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA '12, Pittsburgh, PA, USA, June 25-27, 2012*, pages 193–204. ACM, 2012.

[8] G. Ballard, J. Demmel, O. Holtz, B. Lipshitz, and O. Schwartz. Strong scaling of matrix multiplication algorithms and memory-independent communication lower bounds. 2012.

[9] G. Ballard, J. Demmel, O. Holtz, and O. Schwartz. Graph expansion and communication costs of fast matrix multiplication. *J. ACM*, 59(6):32:1–32:23, 2012.

[10] G. Ballard, C. Ikenmeyer, J. M. Landsberg, and N. Ryder. The geometry of rank decompositions of matrix multiplication II: 3×3 matrices. *CoRR*, abs/1801.00843, 2018.

[11] N. Bansal and R. Williams. Regularity lemmas and combinatorial algorithms. *Theory of Computing*, 8(1):69–94, 2012.

[12] A. R. Benson and G. Ballard. A framework for practical parallel fast matrix multiplication. In A. Cohen and D. Grove, editors, *Proceedings of the 20th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPoPP 2015, San Francisco, CA, USA, February 7-11, 2015*, pages 42–53. ACM, 2015.

[13] D. Bini. Relations between exact and approximate bilinear algorithms. Applications. *Calcolo*, 17:87–97, 1980.

[14] M. Bläser. Fast matrix multiplication. *Theory of Computing, Graduate Surveys*, 5:1–60, 2013.

[15] M. Bläser, M. Christandl, and J. Zuiddam. The border support rank of two-by-two matrix multiplication is seven. *CoRR*, abs/1705.09652, 2017.

[16] H. L. Bodlaender, M. Cygan, S. Kratsch, and J. Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Inf. Comput.*, 243:86–111, 2015.

[17] M. Bodrato. A Strassen-like matrix multiplication suited for squaring and higher power computation. In W. Koepf, editor, *Symbolic and Algebraic Computation, International Symposium, ISSAC 2010, Munich, Germany, July 25-28, 2010, Proceedings*, pages 273–280. ACM, 2010.

[18] R. P. Brent. Algorithms for matrix multiplication. Technical Report STAN-CS-70-157, Stanford University, 1970.

[19] N. H. Bshouty. On the additive complexity of 2×2 matrix multiplication. *Inf. Process. Lett.*, 56(6):329–

335, 1995.

[20] H. Buhrman, M. Christandl, and J. Zuiddam. Nondeterministic quantum communication complexity: the cyclic equality game and iterated matrix multiplication. In C. H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, volume 67 of *LIPIcs*, pages 24:1–24:18. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[21] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic Complexity Theory*. Springer, 1997.

[22] M. Cenk and M. A. Hasan. On the arithmetic complexity of Strassen-like matrix multiplications. *J. Symbolic Comput.*, 80(part 2):484–501, 2017.

[23] T. M. Chan. Speeding up the four Russians algorithm by about one more logarithmic factor. In Indyk [44], pages 212–217.

[24] L. Chiantini, C. Ikenmeyer, J. M. Landsberg, and G. Ottaviani. The geometry of rank decompositions of matrix multiplication I: 2×2 matrices. *Experimental Mathematics*, 2017.

[25] M. Christandl and J. Zuiddam. Tensor surgery and tensor rank. *Comput. Complexity*, 2018.

[26] H. Cohn, R. D. Kleinberg, B. Szegedy, and C. Umans. Group-theoretic algorithms for matrix multiplication. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 379–388. IEEE Computer Society, 2005.

[27] H. Cohn and C. Umans. A group-theoretic approach to fast matrix multiplication. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 438–449. IEEE Computer Society, 2003.

[28] H. Cohn and C. Umans. Fast matrix multiplication using coherent configurations. In S. Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1074–1087. SIAM, 2013.

[29] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *J. Symb. Comput.*, 9(3):251–280, 1990.

[30] M. Cygan, S. Kratsch, and J. Nederlof. Fast Hamiltonicity checking via bases of perfect matchings. *J. ACM*, 65(3):12:1–12:46, 2018.

[31] M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. van Rooij, and J. O. Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In R. Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 150–159. IEEE Computer Society, 2011.

[32] P. D'Alberto, M. Bodrato, and A. Nicolau. Exploiting parallelism in matrix-computation kernels for symmetric multiprocessor systems: Matrix-multiplication and matrix-addition algorithm optimizations by software pipelining and threads allocation. *ACM Trans. Math. Softw.*, 38(1):2:1–2:30, 2011.

[33] A. M. Davie and A. J. Stothers. Improved bound for complexity of matrix multiplication. *Proc. R. Soc. Edinb., Sect. A, Math.*, 143(2):351–369, 2013.

[34] P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices I: approximating matrix multiplication. *SIAM J. Comput.*, 36(1):132–157, 2006.

[35] M. J. Fischer and A. R. Meyer. Boolean matrix multiplication and transitive closure. In *12th Annual Symposium on Switching and Automata Theory, East Lansing, Michigan, USA, October 13-15, 1971*, pages 129–131. IEEE Computer Society, 1971.

[36] F. V. Fomin, D. Lokshtanov, F. Panolan, and S. Saurabh. Efficient computation of representative families with applications in parameterized and exact algorithms. *J. ACM*, 63(4):29:1–29:60, 2016.

[37] M. E. Furman. Application of a method of fast multiplication of matrices in the problem of finding the transitive closure of a graph. *Sov. Math., Dokl.*, 11:1252, 1970.

[38] B. Grayson and R. A. van de Geijn. A high performance parallel strassen implementation. *Parallel Processing Letters*, 6(1):3–12, 1996.

[39] J. Håstad. Tensor rank is NP-complete. *J. Algorithms*, 11(4):644–654, 1990.

[40] C. J. Hillar and L.-H. Lim. Most tensor problems are NP-hard. *J. ACM*, 60(6):Art. 45, 39, 2013.

[41] J. E. Hopcroft and L. R. Kerr. On minimizing the number of multiplications necessary for matrix multiplication. *SIAM J. Appl. Math.*, 20:30–36, 1971.

[42] J. Huang, L. Rice, D. A. Matthews, and R. A. van de Geijn. Generating families of practical fast matrix multiplication algorithms. In *2017 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2017, Orlando, FL, USA, May 29 - June 2, 2017*, pages 656–667. IEEE Computer Society, 2017.

[43] J. Huang, T. M. Smith, G. M. Henry, and R. A. van de Geijn. Strassen's algorithm reloaded. In J. West and C. M. Pancake, editors, *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2016, Salt Lake City, UT, USA, November 13-18, 2016*, pages 690–701. IEEE Computer Society, 2016.

[44] P. Indyk, editor. *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*. SIAM, 2015.

[45] A. Itai and M. Rodeh. Finding a minimum circuit in a graph. *SIAM J. Comput.*, 7(4):413–423, 1978.

[46] S. Jukna and I. Sergeev. Complexity of linear Boolean operators. *Foundations and Trends in Theoretical Computer Science*, 9(1):1–123, 2013.

[47] I. Kaporin. A practical algorithm for faster matrix multiplication. *Numer. Linear Algebra Appl.*, 6(8):687–700, 1999.

[48] I. Kaporin. The aggregation and cancellation tech-

niques as a practical tool for faster matrix multiplication. *Theoret. Comput. Sci.*, 315(2-3):469–510, 2004.

[49] E. Karstadt and O. Schwartz. Matrix multiplication, a little faster. In C. Scheideler and M. T. Hajiaghayi, editors, *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2017, Washington DC, USA, July 24-26, 2017*, pages 101–110. ACM, 2017.

[50] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009.

[51] B. Kumar, C. Huang, R. W. Johnson, and P. Sadayappan. A tensor product formulation of strassen's matrix multiplication algorithm with memory reduction. In *The Seventh International Parallel Processing Symposium, Proceedings, Newport Beach, California, USA, April 13-16, 1993.*, pages 582–588. IEEE Computer Society, 1993.

[52] J. M. Landsberg. The border rank of the multiplication of 2×2 matrices is seven. *J. Am. Math. Soc.*, 19(2):447–459, 2006.

[53] J. M. Landsberg. *Tensors: Geometry and Applications*, volume 128 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2012.

[54] J. M. Landsberg and M. Michałek. On the geometry of border rank decompositions for matrix multiplication and other tensors with symmetry. *SIAM J. Appl. Algebra Geom.*, 1(1):2–19, 2017.

[55] J. M. Landsberg and N. Ryder. On the geometry of border rank algorithms for $n \times 2$ by 2×2 matrix multiplication. *Exp. Math.*, 26(3):275–286, 2017.

[56] F. Le Gall. Powers of tensors and fast matrix multiplication. In K. Nabeshima, K. Nagasaka, F. Winkler, and Á. Szántó, editors, *International Symposium on Symbolic and Algebraic Computation, ISSAC '14, Kobe, Japan, July 23-25, 2014*, pages 296–303. ACM, 2014.

[57] B. Lipshitz, G. Ballard, J. Demmel, and O. Schwartz. Communication-avoiding parallel Strassen: implementation and performance. In J. K. Hollingsworth, editor, *SC Conference on High Performance Computing Networking, Storage and Analysis, SC '12, Salt Lake City, UT, USA - November 11 - 15, 2012*, page 101. IEEE/ACM, 2012.

[58] Q. Luo and J. B. Drake. A scalable parallel Strassen's matrix multiplication algorithm for distributed-memory computers. In J. Hightower, E. Deaton, K. M. George, J. H. Carroll, and D. Oppenheim, editors, *Proceedings of the 1995 ACM symposium on applied computing, SAC'95, Nashville, TN, USA, February 26-28, 1995*, pages 221–226. ACM, 1995.

[59] R. Pagh. Compressed matrix multiplication. *TOCT*, 5(3):9:1–9:17, 2013.

[60] V. Pan. How can we speed up matrix multiplication? *SIAM Rev.*, 26(3):393–415, 1984.

[61] V. Y. Pan. Strassen's algorithm is not optimal: Trililnear technique of aggregating, uniting and canceling for constructing fast algorithms for matrix operations. In *19th Annual Symposium on Foundations of Computer Science, Ann Arbor, Michigan, USA, 16-18 October 1978* [1], pages 166–176.

[62] V. Y. Pan. Fast feasible and unfeasible matrix multiplication. *CoRR*, abs/1804.04102, 2018.

[63] V. Y. Pan, Q. Luan, J. Svadlenka, and L. Zhao. Superfast accurate low rank approximation. *CoRR*, abs/1710.07946, 2017.

[64] R. L. Probert. On the additive complexity of matrix multiplication. *SIAM J. Comput.*, 5(2):187–203, 1976.

[65] A. A. Razborov. Lower bounds on the dimension of schemes of bounded depth in a complete basis containing the logical addition function. *Mat. Zametki*, 41(4):598–607, 623, 1987.

[66] A. Schönhage. Partial and total matrix multiplication. *SIAM J. Comput.*, 10(3):434–455, 1981.

[67] A. V. Smirnov. The bilinear complexity and practical algorithms for matrix multiplication. *Zh. Vychisl. Mat. Mat. Fiz.*, 53(12):1970–1984, 2013.

[68] R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In A. V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 77–82. ACM, 1987.

[69] V. Strassen. Gaussian elimination is not optimal. *Numer. Math.*, 13:354–356, 1969.

[70] V. Strassen. Relative bilinear complexity and matrix multiplication. *J. Reine Angew. Math.*, 375/376:406–443, 1987.

[71] L. G. Valiant. General context-free recognition in less than cubic time. *J. Comput. Syst. Sci.*, 10(2):308–315, 1975.

[72] V. Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. In H. J. Karloff and T. Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 887–898. ACM, 2012.

[73] V. Vassilevska Williams and R. Williams. Subcubic equivalences between path, matrix and triangle problems. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 645–654. IEEE Computer Society, 2010.

[74] R. Williams. Faster all-pairs shortest paths via circuit complexity. In D. B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 664–673. ACM, 2014.

[75] S. Winograd. On multiplication of 2×2 matrices. *Linear Algebra Appl.*, 4:381–388, 1971.

[76] H. Yu. An improved combinatorial algorithm for Boolean matrix multiplication. *Inf. Comput.*, 261:240–247, 2018.